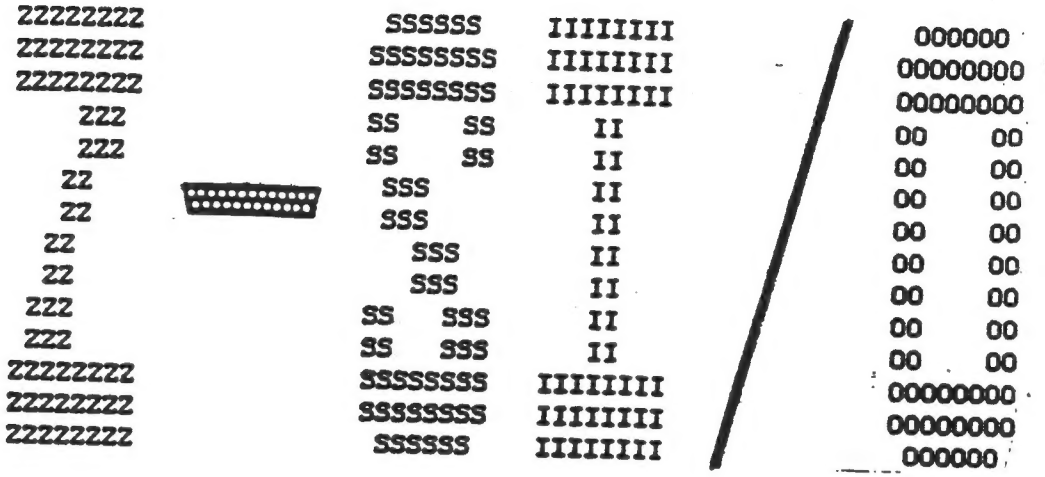


Z-SI/O SERIAL INTERFACE MODULE

INSTALLATION / PROGRAMMING
AND
TECHNICAL MANUAL

****

JLO DISK VERSION

SEP 18 86

SEE PAGE 20

0007

TABLE OF CONTENTS

CLIFFORD & Associates

Z-SI/O SERIAL INPUT/OUTPUT MODULE

INTRODUCTION

This manual supplies the information needed to install, test and use the Z-SI/O Serial Input/Output Module. We suggest that you first scan the entire manual before starting.

Should you encounter any problem during installation call on us for help if necessary. If your module does not work properly, recheck your installation step by step. Once you are satisfied that the module is correctly installed, feel free to ask for our help.

GENERAL INFORMATION

Z-SI/O Description

The Z-SI/O Serial Input/Output Module is a "hardware based" serial interface which allows the Timex Sinclair 2068 Personal Color Computer to support the EIA RS-232C standards applicable to the 25 pin interconnection of Data Terminal Equipment (DTE) and Data Communication Equipment (DCE) employing serial binary data interchange.

The "stock" Z-SI/O boardlet configures the TS-2068 as Data Terminal Equipment and, with the appropriate USER SUPPLIED software, will allow the TS-2068 to communicate with any RS-232C modem, serial printer or serial X-Y plotter. The "stock" Z-SI/O supports 2 basic baud rates (300 & 1200) 5, 6, 7 and 8 bit character lengths, EVEN/ODD or O parity and 1, 1.5 or 2 stop bits all through simple software commands. The module I/O addresses are "etch-jumpered" for using the Z-SI/O at the following port addresses:- DATA= 9Fh and CONTROL/STATUS= BFh.

Receiving Inspection

When your module arrives, examine the shipping container for signs of possible damage to the contents during transit. Then inspect the contents for damage. (We suggest you save the shipping materials for use in returning the module to CLIFFORD & Associates should it become necessary to do so.) If your Z-SI/O is damaged, please write us at once describing the condition so that we can take appropriate action.

Z-SI/O SERIAL INPUT/OUTPUT MODULE

INSTALLATION

Because the Z-SI/O Interface Card derives it's power from the TS-2068's 15Volt supply to power the RS-232C line drivers, the Z-SI/O Card should be the "FIRST" device plugged into the TS-2068's Expansion Port.

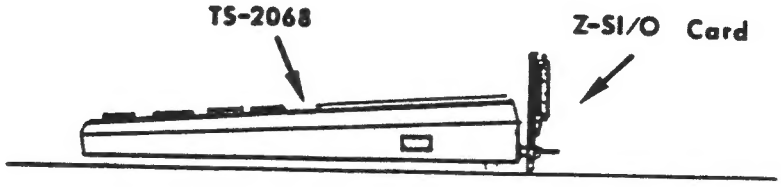
Any other peripherals like the 2040 Printer, 2050 Modem, Z-Link Interface and others may be plugged into the rear of the Z-SI/O Card. Infact the Z-SI/O becomes an 'extention' of the TS-2068's Expansion Port.... ALL OF THE BUSS LINES ARE PASSED THROUGH AND ARE AVAILABLE ON THE REAR OF THE Z-SI/O CARD.

To install the Z-SI/O Card, TURN-OFF ALL POWER TO THE COMPUTER AND ANY OTHER PERIPHERALS THAT MAY BE CONNECTED...

DISCONNECT any peripherals from the TS-2068's Expansion Port.

CAREFULLY align the card-edge connector on the Z-SI/O Card to that of the TS-2068's Expansion Port. You should "feel" the index tab in the Z-SIO's connector "catch" when it is aligned with the index slot in the TS-2068's Expansion Port edge connector. When this happens GENTLY press the Z-SI/O Card on the connector. Use enough force to FULLY seat the connector.

If properly fitted, the installation should look like that shown in the illustration below.



** CAUTION **

THE CONNECTOR MAY FEEL A LITTLE 'SNUG' GOING ON. BE SURE THE Z-SI/O CARD IS PROPERLY ALIGNED AND FULLY SEATED BEFORE APPLYING POWER TO THE COMPUTER!!!
FAILURE TO TO DO SO MAY RESULT IN PERMANENT DAMAGE TO THE TS-2068 AND THE Z-SI/O CARD!!

BEFORE re-connecting any of your other peripherals, turn ON the POWER to the TS-2068. It should 'BOOT-UP' and display the Timex/SINCLAIR 'copyright' as normal.
If all is well you may, after shutting OFF the power, re-connect the rest of your peripherals and proceed with connecting your RS-232 device and begin writing application software or if you purchased the optional 'SPECTERM-64' terminal software, follow the MODEM manufacture's instructions for installing and operating your Modem.

PROGRAMMING

A set of control words must be sent to the Z-SI/O Module to define the desired mode and communications format. The control words will specify the BAUD rate factor (1x, 16x, 64x), character length (5 to 8), number of STOP bits (1, 1-1/2, 2) Asynchronous or Synchronous mode, SYNDET (IN or OUT), parity, etc.

The "heart" of the Z-SI/O Module is the Intel 8251 Universal Synchronous/Asynchronous Reciever/Transmitter (USART for short) and is designed for microcomputer systems data communications. The USART is used as a peripheral and is programmed by the host systems processor (in this case the Z-80 CPU in the TS-2068). The following listings are examples of programming the Z-SI/O for various modes of operation. (FOR A MORE DETAILED DISCUSSION REFER TO THE 8251 DATA SHEET IN THE APPENDIX OF THIS MANUAL.)

Z-SI/O Init routines in Z-80 ASSEMBLY Language....

```
  Object    Source
   code     statement

   3E00     LD   A,0
   D3BF     OUT  (191),A
   3E00     LD   A,0          Clears the 8251 and is usually
   D3BF     OUT  (191),A      preformed after a 'cold' system
   3E00     LD   A,0          "power-up".
   3DBF     OUT  (191),A

   3E40     LD   A,64         Resets the 8251 USART to the
   D3BF     OUT  (191),A      "MODE INSTRUCTION" command level.

   3E6F     LD   A,111        "MODE INSTRUCTION" The '111' sets
   D3BF     OUT  (191),A      the Z-SIO for 300 BAUD, 0 PARITY,
                              8 BIT WORD and 1 STOP BIT.
                              (For 1200 BAUD, change '111' to
                              '110')

   3EB7     LD   A,183        "COMMAND INSTRUCTION"  '183' sets
   D3BF     OUT  (191),A      DTR & RTS low, enables transmitter
                              and receiver, resets (PE, OE, FE)
                              error flags.

   C9       RET    <--------  A RETurn instruction, could also be
                              a "JUMP" to another portion of your
                              program.
```

Now that the USART is initalized you may send/receive DATA. The folowing EXAMPLE routines show how to OUTPUT data and test the STATUS flags of the reciver and transmitter as well as INPUT data using the 'A' accumulator....

To OUTPUT data....

```
LD   A,data     ;Put DATA in 'A'
OUT  (159),A    ;Send it OUT the data port
RET             ;Return
```

To INPUT data...

```
IN   A,(159)    ;Get DATA from data port and
                ;put it in the 'A' register
RET             ;and RETurn to process it.
```

To get transmitter STATUS...

```
IN   A,(191)    ;Get STATUS form USART
BIT  0,A        ;Mask BIT to be tested <-- THIS CAN BE BIT 2
RET             ;and RETurn to process it.
```

To get receiver STATUS...

```
IN   A,(191)    ;Get STATUS from USART
BIT  1,A        ;Mask BIT to be tested
RET             ;and RETurn to process it.
```

The following is an EXAMPLE routine written in BASIC that will initialize the Z-SIO for an 8 BIT WORD, 0 PARITY, 1 STOP BIT and 300 BAUD...

```
10   OUT 191,0
15   OUT 191,0
20   OUT 191,0
25   OUT 191,64
30   OUT 191,111 <---(For 1200 Baud, change '111' to '110')
35   OUT 191,183
```

The above routine is the BASIC equivelent to the one written in Z-80 code on the previous page.

Writing ACTUAL software drivers and program 'patches' are far beyond the scope of this manual. We hope however that with the inclusion of the EXAMPLE routines shown below that they may help you in the writing of your own programs and software patches.

The first is an EXAMPLE of sending both, numerical and variable-length character STRINGS to a serial printer (we used an OKIDATA Microline 93a set for 8/1/0 at 1200 Baud and "printer BUSY" line tied to Z-SIO's 'CTS' line).

Here is the BASIC routine...

```
  1  OUT 191,0:OUT 191,0:OUT 191,0 <--- Clears USART.
  2  OUT 191,64            <--- Sets USART to accept MODE command.
  3  OUT 191,110           <--- Sets USART for 8/1/0 at 1200 baud.
  4  OUT 191,183           <--- Enables TX/RX, sets DTR, RTS low.
  5  OUT 159,13            <--- Sends a "hard" carriage return.
  6  PAUSE 30
100  GOTO 300              <--- Go get data to be transmitted.
120  LET X$=STR$ X         <--- Set-up the string.
130  FOR Y=1 TO LEN X$     <--- Set loop counter.
140  LET stat=IN 191       <--- Get USART status.
150  IF stat-INT (stat/2)*2=0 THEN GOTO 140  <-- Is it ready?
170  OUT 159,CODE (X$(Y))  <--- Go send a byte.
180  PAUSE 3
190  NEXT Y          <--- Do it again.
200  OUT 159,13      <--- Sends a C/R.
202  PAUSE 10
205  OUT 159,10      <--- Sends a linefeed (L/F).
220  RETURN
300  LET X=2068      <--- A numerical string to be printed.
310  GOSUB 120       <--- Go print it.
350  OUT 159,13      <--- C/R
352  PAUSE 10
355  OUT 159,10      <--- L/F
400  LET X=10        <--- Another 'n' string.
410  GOSUB 120       <--- Go print it.
500  LET X$="Z-SI/O RS-232c Interface." <- A 'c' string to print
510  GOSUB 130       <--- Go print it.
520  LET X$="This is a TEST at 1200 BAUD on my serial printer."
550  GOSUB 130       <--- Go print it.
600  LET X=30*2      <--- Another 'n' string.
610  GOSUB 120       <--- Go print the answer.
615  PRINT "All DONE!"
```

The next EXAMPLE is a VERY simple terminal routine that may be used to transmit and recieve ASCII characters with other RS-232 devices such as a terminal or a second computer and shows a second approach as to how to test the Z-SIO STATUS port using a BIN statement to 'mask' the bit to be tested.

```
'set-up USART
 10 OUT 191,0: OUT 191,0: OUT 191,0  <-- 'clear' USART
 11 OUT 191,64  <-- 'tell' USART next code is MODE command.
 12 OUT 191,111 <-- Set for 8/1/0 and X64 (300 Baud)
 13 OUT 191,183 <-- Set ERROR flags, enable TX & RX
'
'clear junk from USART
 14 LET g=IN 159    <-- This stuff is 'optional' but it gets any
 15 OUT 159,0       <-- garbage the USART may have in it's TX &
 16 OUT 159,0       <-- RX registers.
'
'main terminal routine
 50 LET st=IN 191                              <-- Get status
100 IF st<=BIN 10010101 THEN GOTO 200          <-- Is it what we want?
150 IF IN 159 < 32 THEN GOTO 50                <-- If not, get another.
155 PRINT CHR$ IN 159;: POKE 23692,50          <-- PRINT it, POKE scroll.
200 LET t=CODE INKEY$                          <-- Check keyboard for input.
250 IF t=0 THEN GOTO 50                        <-- If nothing, check RX port.
300 IF IN 191=BIN 10010101 THEN GOTO 400       <-- Check if TX ready.
350 GOTO 50                                    <-- If not, check RX port.
400 OUT 159,t                                  <-- Send character.
450 GOTO 50                                    <-- Go do it all again.
```

If the device you are sending to does not 'echo' the character back to you and you wish to "see" what you are sending you could add a line like this:

```
425 PRINT CHR$(t);
```

Also this routine is very 'basic' and is not recomended for accessing an "on-line" system like a "BBS" or "COMPUSERVE" and the BASIC on the TS-2068 is too slow for 1200 Baud. Even at 300 Baud, and if the terminal routine was properly 'complimented' to process CONTROL codes, you will begin to experience 'lost characters' and other strange things.

OPTIONAL TELECOMMUNICATIONS SOFTWARE:-

SPECTERM - 64 Version 3.0

NOTE: Version 3.0 is for the TS-2068 running in 'SPECTRUM MODE' with ANY SPECTRUM ROM, ROM SWITCH or 'EMULATOR'.

Allows the Timex/SINCLAIR 2068 fitted with the Z-SI/O Interface Card to use most ANY RS-232C 300/1200 Baud MODEM to communicate with remote databases, RCP/Ms (Bulletin Board Systems) or other computer systems.

A FULLY implimented communications software package, SPECTERM-64 has many features not found in any other commercial telecommunications software package available for the Timex/Sinclair Computer.

It occupies LESS than 4K of RAM but don't let it's 'small' size fool you.....
Here is a small sample of built-in features...

- o Full 64 column terminal display
- o XMODEM protocol file transfer
- o Up to 35.5K Buffer capacity
- o 300/1200 Baud support
- o Allows use of Z-SI/O with ANY RS-232C Modem
- o Lower/UPPER case letters with "LOCK"
- o "Open-Architecture" - SPECTERM-64 is easy to modify and customize to a user's particular needs.
- o 'BASIC' driven "LOCAL COMMAND MENU" allows YOU to customize the local command structure to operate with YOUR mass storage system.
- o SUPPORT! Refinements to SPECTERM are underway all the time and there is even 24 Hour 'ON-LINE' support for registered owners of SPECTERM-64 software.....

FOR PRICES AND AVAILABILITY, CONTACT YOUR FAVORITE TIMEX-SINCLAIR SOFTWARE DEALER OR WRITE:

G&C COMPUTER PRODUCTS
P.O. Box 2186
Inglewood, California
90305

CLIFFORD & Associates
Z-SI/O RS-232C SERIAL INPUT/OUTPUT MODULE & INTERFACE CARD
DISCLAIMER AND WARRANTY SERVICE NOTICE

****

N O T I C E !

The information provided herein is believed to be reliable; however, CLIFFORD & Associates assumes no responsibility for the use of this information, and all use of such information shall be entirely at the user's own risk. Prices and specifications are subject to change without notice. No patent rights or licenses to any of the circuits described herein are implied or granted to any third party. CLIFFORD & Associates does not authorize or warrant any CLIFFORD & Associates product for use in life support devices and/or systems.

90 Day WARRANTY

FOR A PERIOD OF 90 DAYS AFTER DATE OF PURCHASE....
If in the event that this product fails to perform due to a defect in materials or workmanship, CLIFFORD & Associates will (at our option) repair or replace it if it is returned, insured and shipping prepaid with proof of purchase to:

CLIFFORD & Associates
13910 Halldale Avenue
Gardena, California
90249

CLIFFORD & Associates will not be liable for any injury, loss, or damage, direct or consequental, arising out of the use of, or the inability to use, this product.
The adaptability of the Z-SI/O Interface Card for the use with a particular peripheral is the sole responsibility of the Purchaser/User.

OUT OF WARRANTY REPAIRS

For OUT-OF-WARRANTY service or replacement parts, please write to us for the latest price quotes.

SPECIAL SERVICES... Need more than one RS-232 port or a 'custom' address or a 'non-standard' baud rate?

Write to us with your requirements in detail...
Our rates are reasonable for custom configurations!

****

PROGRAMMABLE COMMUNICATION INTERFACES

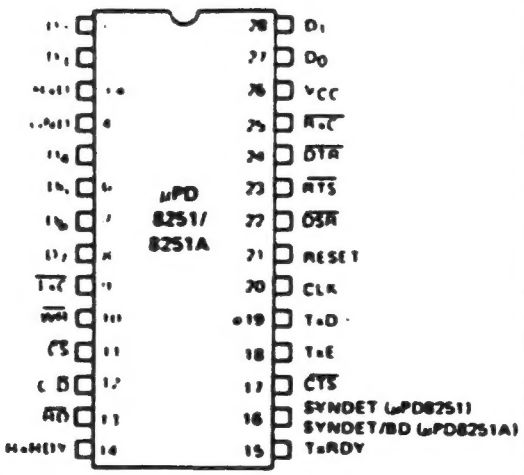
DESCRIPTION

The μPD8251 and μPD8251A Universal Synchronous/Asynchronous Receiver/Transmitters (USARTs) are designed for microcomputer systems data communications. The USART is used as a peripheral and is programmed by the 8080A or other processor to communicate in commonly used serial data transmission techniques including IBM Bi-Sync. The USART receives serial data streams and converts them into parallel data characters for the processor. While receiving serial data, the USART will also accept data characters from the processor in parallel format, convert them to serial format and transmit. The USART will signal the processor when it has completely received or transmitted a character and requires service. Complete USART status including data format errors and control signals such as TxE and SYNDET, is available to the processor at any time.

FEATURES

- Asynchronous or Synchronous Operation
  - Asynchronous
    - Five 8-Bit Characters
    - Clock Rate – 1, 16 or 64 x Baud Rate
    - Break Character Generation
    - Select 1, 1-1/2, or 2 Stop Bits
    - False Start Bit Detector
    - Automatic Break Detect and Handling (μPD8251A)
  - Synchronous
    - Five 8 Bit Characters
    - Internal or External Character Synchronization
    - Automatic Sync Insertion
    - Single or Double Sync Characters
- Baud Rate (1X Mode) – DC to 56K Baud (μPD8251)
  – DC to 64K Baud (μPD8251A)
- Full Duplex, Double Buffered Transmitter and Receiver
- Parity, Overrun and Framing Flags
- Fully Compatible with 8080A/8085/μPD780 (Z80™)
- All Inputs and Outputs are TTL Compatible
- Single +5 Volt Supply, ±10%
- Separate Device Receive and Transmit TTL Clocks
- 28 Pin Plastic DIP Package
- N Channel MOS Technology

PIN CONFIGURATION



PIN NAMES

D_7-D_0	Data Bus (8 bits)
C/D	Control or Data is to be Written or Read
RD	Read Data Command
WR	Write Data or Control Command
CS	Chip Enable
CLK	Clock Pulse (TTL)
RESET	Reset
TxC	Transmitter Clock (TTL)
TxD	Transmitter Data
RxC	Receiver Clock (TTL)
RxD	Receiver Data
RxRDY	Receiver Ready (has character for 8080A)
TxRDY	Transmitter Ready (ready for char. from 8080A)
DSR	Data Set Ready
DTR	Data Terminal Ready
SYNDET	Sync Detect
SYNDET/BD	Sync Detect/Break Detect
RTS	Request to Send Data
CTS	Clear to Send Data
TxE	Transmitter Empty
V_{CC}	+5 Volt Supply
GND	Ground

TM. Z80 is a registered trademark of Zilog.

FUNCTIONAL DESCRIPTION

The μPD8251 and μPD8251A Universal Synchronous/Asynchronous Receiver/Transmitters are designed specifically for 8080 microcomputer systems but work with most 8-bit processors. Operation of the μPD8251 and μPD8251A, like other I/O devices in the 8080 family, are programmed by system software for maximum flexibility.

In the receive mode, the μPD8251 or μPD8251A converts incoming serial format data into parallel data and makes certain format checks. In the transmit mode, it formats parallel data into serial form. The device also supplies or removes characters or bits that are unique to the communication format in use. By performing conversion and formatting services automatically, the USART appears to the processor as a simple or "transparent" input or output of byte-oriented parallel data.

μPD8251A FEATURES AND ENHANCEMENTS

The μPD8251A is an advanced design of the industry standard 8251 USART. It operates with a wide range of microprocessors, including the 8080, 8085, and μPD780 (Z80™). The additional features and enhancements of the μPD8251A over the μPD8251 are listed below.

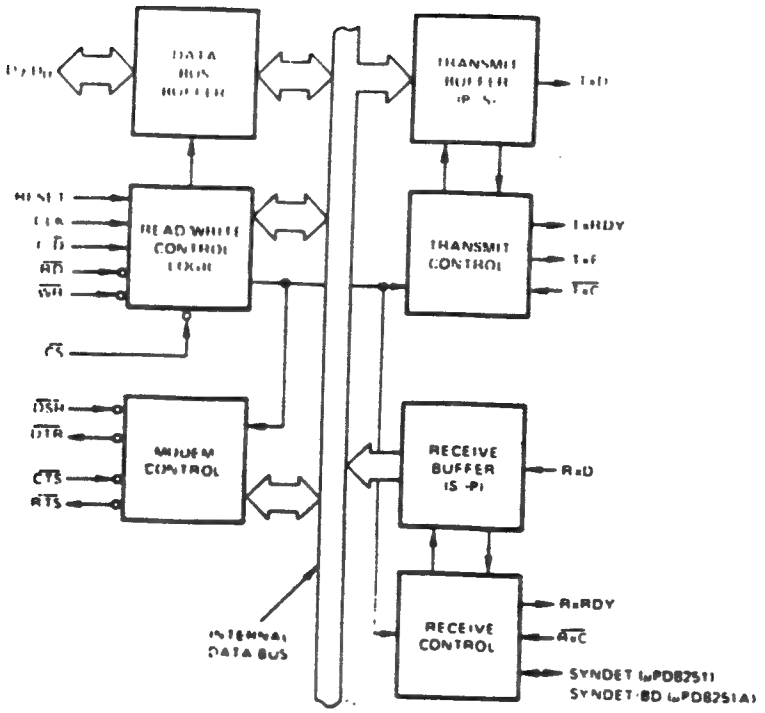
1. The data paths are double-buffered with separate I/O registers for control, status, Data In and Data Out. This feature simplifies control programming and minimizes processor overhead.
2. The Receiver detects and handles "break" automatically in asynchronous operations, which relieves the processor of this task.
3. The Receiver is prevented from starting when in "break" state by a refined Rx initialization. This also prevents a disconnected USART from causing unwanted interrupts.
4. When a transmission is concluded the TxD line will always return to the marking state unless SBRK is programmed.
5. The Tx Disable command is prevented from halting transmission by the Tx Enable Logic enhancement, until all data previously written has been transmitted. The same logic also prevents the transmitter from turning off in the middle of a word.
6. Internal Sync Detect is disabled when External Sync Detect is programmed. An External Sync Detect Status is provided through a flip-flop which clears itself upon a status read.
7. The possibility of a false sync detect is minimized by:
   - ensuring that if a double sync character is programmed, the characters be contiguously detected.
   - clearing the Rx register to all Logic 1s (V_{OH}) whenever the Enter Hunt command is issued in Sync mode.
8. The $\overline{RD}$ and $\overline{WR}$ do not affect the internal operation of the device as long as the μPD8251A is not selected.
9. The μPD8251A Status can be read at any time, however, the status update will be inhibited during status read.
10. The μPD8251A has enhanced AC and DC characteristics and is free from extraneous glitches, providing higher speed and improved operating margins.
11. Baud rate from DC to 64K.

BASIC OPERATION

C/$\overline{D}$	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	
0	0	1	0	μPD8251/μPD8251A → Data Bus
0	1	0	0	Data Bus → μPD8251/μPD8251A
1	0	1	0	Status → Data Bus
1	1	0	0	Data Bus → Control
X	X	X	1	Data Bus → 3-State
X	1	1	0	

TM Z80 is a registered trademark of Zilog.

BLOCK DIAGRAM



ABSOLUTE MAXIMUM RATINGS*

Operating Temperature	−0°C to +70°C
Storage Temperature	−65°C to +125°C
All Output Voltages	−0.5 to +7 Volts
All Input Voltages	−0.5 to +7 Volts
Supply Voltages	−0.5 to +7 Volts

COMMENT Stress above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

DC CHARACTERISTICS

*T_a 25°C

T_a = 0°C to 70°C V_{CC} 5.0V ± 10% GND 0V

PARAMETER	SYMBOL	LIMITS μPD8251 MIN	TYP	MAX	LIMITS μPD8251A MIN	MAX	UNIT	TEST CONDITIONS
Input Low Voltage	V_{IL}	-0.5		0.8	0.5	0.8	V	
Input High Voltage	V_{IH}	2.0		V_{CC}	2.0	V_{CC}	V	
Output Low Voltage	V_{OL}			0.45		0.45	V	μPD8251 I_{OL} = 1.7 mA μPD8251A I_{OL} = 2.2 mA
Output High Voltage	V_{OH}	2.4			2.4		V	μPD8251 I_{OH} = −100 μA μPD8251A I_{OH} = −400 μA
Data Bus Leakage	I_{DL}			-50		-10	μA	V_{OUT} = 0.45V
				10		10		V_{OUT} = V_{CC}
Input Load Current	I_{IL}			10		10	μA	At 5.5V
Power Supply Current	I_{CC}		45	80		100	mA	μPD8251A All Outputs = Logic 1

CAPACITANCE

T_a 25°C V_{CC} GND 0V

PARAMETER	SYMBOL	LIMITS MIN	TYP	MAX	UNIT	TEST CONDITIONS
Input Capacitance	C_{IN}	—		10	pF	fc = 1 MHz
I/O Capacitance	$C_{I/O}$			20	pF	Unmeasured pins returned to GND

PIN IDENTIFICATION

PIN NO.	PIN SYMBOL	PIN NAME	FUNCTION
1, 2, 27, 28 5 – 8	$D_7 - D_0$	Data Bus Buffer	An 8-bit, 3-state bi-directional buffer used to interface the USART to the processor data bus. Data is transmitted or received by the buffer in response to input/output or Read/ Write instructions from the processor. The Data Bus Buffer also transfers Control words, Command words, and Status.
26	V_{CC}	V_{CC} Supply Voltage	+5 volt supply
4	GND	Ground	Ground
Read/Write Control Logic			This logic block accepts inputs from the processor Control Bus and generates control signals for overall USART operation. The Mode Instruction and Command Instruction registers that store the control formats for device functional definition are located in the Read/ Write Control Logic.
21	RESET	Reset	A "one" on this input forces the USART into the "idle" mode where it will remain until reinitialized with a new set of control words. Minimum RESET pulse width is 6 t_{CY}.
20	CLK	Clock Pulse	The CLK input provides for internal device timing and is usually connected to the Phase 2 (TTL) output of the μPB8224 Clock Generator. External inputs and outputs are not referenced to CLK, but the CLK frequency must be at least 30 times the Receiver or Transmitter clocks in the synchronous mode and 4.5 times for the asynchronous mode
10	$\overline{WR}$	Write Data	A "zero" on this input instructs the USART to accept the data or control word which the processor is writing out on the data bus
13	$\overline{RD}$	Read Data	A "zero" on this input instructs the USART to place the data or status information onto the Data Bus for the processor to read
12	C/$\overline{D}$	Control/Data	The Control/Data input, in conjunction with the $\overline{WR}$ and $\overline{RD}$ inputs, informs the USART to accept or provide either a data character, control word or status information via the Data Bus 0 = Data, 1 = Control
11	$\overline{CS}$	Chip Select	A "zero" on this input enables the USART to read from or write to the processor
Modem Control			The μPD8251 and μPD8251A have a set of control inputs and outputs which may be used to simplify the interface to a Modem
22	$\overline{DSR}$	Data Set Ready	The Data Set Ready input can be tested by the processor via Status information. The $\overline{DSR}$ input is normally used to test Modem Data Set Ready condition
24	$\overline{DTR}$	Data Terminal Ready	The Data Terminal Ready output can be controlled via the Command word. The $\overline{DTR}$ output is normally used to drive Modem Data Terminal Ready or Rate Select lines
23	$\overline{RTS}$	Request to Send	The Request to Send output can be controlled via the Command word. The $\overline{RTS}$ output is normally used to drive the Modem Request to Send line
17	$\overline{CTS}$	Clear to Send	A "zero" on the Clear to Send input enables the USART to transmit serial data if the TxEN bit in the Command Instruction register is enabled (one)

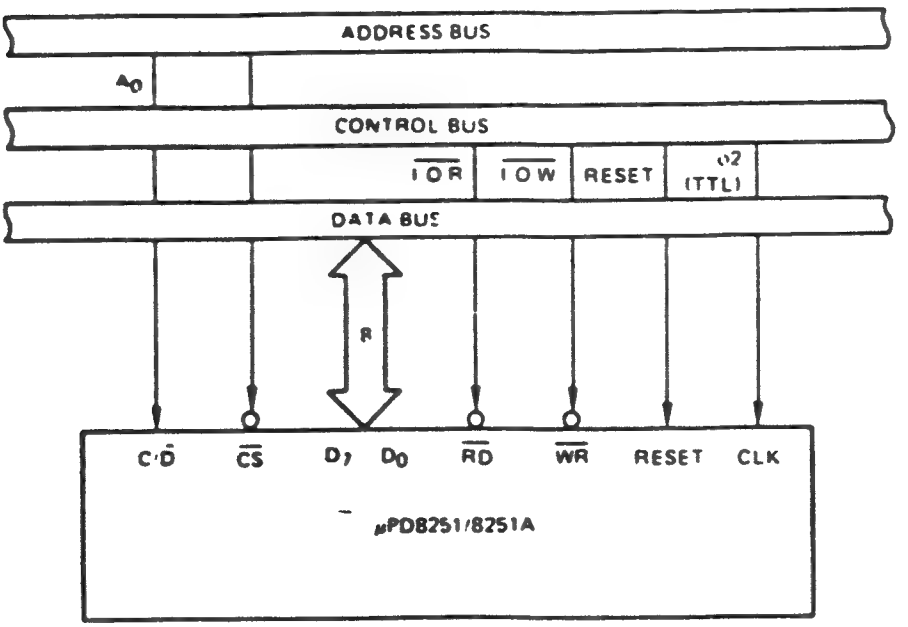
TRANSMIT BUFFER

The Transmit Buffer receives parallel data from the Data Bus Buffer via the internal data bus, converts parallel to serial data, inserts the necessary characters or bits needed for the programmed communication format and outputs composite serial data on the TxD pin.

PIN IDENTIFICATION (CONT.)

PIN NO.	PIN SYMBOL	PIN NAME	FUNCTION
Transmit Control Logic			The Transmit Control Logic accepts and outputs all external and internal signals necessary for serial data transmission.
15	TxRDY	Transmitter Ready	Transmitter Ready signals the processor that the transmitter is ready to accept a data character. TxRDY can be used as an interrupt or may be tested through the Status information for polled operation. Loading a character from the processor automatically resets TxRDY, on the leading edge.
18	TxE	Transmitter Empty	The Transmitter Empty output signals the processor that the USART has no further characters to transmit. TxE is automatically reset upon receiving a data character from the processor. In half-duplex, TxE can be used to signal end of a transmission and request the processor to "turn the line around." The TxEn bit in the command instruction does not effect TxE. In the Synchronous mode, a "one" on this output indicates that a Sync character or characters are about to be automatically transmitted as "fillers" because the next data character has not been loaded.
9	$\overline{TxC}$	Transmitter Clock	The Transmitter Clock controls the serial character transmission rate. In the Asynchronous mode, the $\overline{TxC}$ frequency is a multiple of the actual Baud Rate. Two bits of the Mode Instruction select the multiple to be 1x, 16x, or 64x the Baud Rate. In the Synchronous mode, the $\overline{TxC}$ frequency is automatically selected to equal the actual Baud Rate Note that for both Synchronous and Asynchronous modes, serial data is shifted out of the USART by the falling edge of $\overline{TxC}$.
19	TxD	Transmitter Data	The Transmit Control Logic outputs the composite serial data stream on this pin.

μPD8251 AND μPD8251A INTERFACE TO 8080 STANDARD SYSTEM BUS



RECEIVE BUFFER

The Receive Buffer accepts serial data input at the $\overline{RxD}$ pin and converts the data from serial to parallel format. Bits or characters required for the specific communication technique in use are checked and then an eight-bit "assembled" character is readied for the processor. For communication techniques which require less than eight bits, the μPD8251 and μPD8251A set the extra bits to "zero."

PIN IDENTIFICATION (CONT.)

PIN NO.	PIN SYMBOL	PIN NAME	FUNCTION
Receiver Control Logic			This block manages all activities related to incoming data.
14	RxRDY	Receiver Ready	The Receiver Ready output indicates that the Receiver Buffer is ready with an "assembled" character for input to the processor. For Polled operation, the processor can check RxRDY using a Status Read or RxRDY can be connected to the processor interrupt structure. Note that reading the character to the processor automatically resets RxRDY.
25	$\overline{RxC}$	Receiver Clock	The Receiver Clock determines the rate at which the incoming character is received. In the Asynchronous mode, the $\overline{RxC}$ frequency may be 1.16 or 64 times the actual Baud Rate but in the Synchronous mode the $\overline{RxC}$ frequency must equal the Baud Rate. Two bits in the mode instruction select Asynchronous at 1x, 16x or 64x or Synchronous operation at 1x the Baud Rate. Unlike $\overline{TxC}$, data is sampled by the μPD8251 and μPD8251A on the rising edge of $\overline{RxC}$. ①
3	RxD	Receiver Data	A composite serial data stream is received by the Receiver Control Logic on this pin.
16	SYNDET (μPD8251)	Sync Detect	The SYNC Detect pin is only used in the Synchronous mode. The μPD8251 may be programmed through the Mode Instruction to operate in either the internal or external Sync mode and SYNDET then functions as an output or input respectively. In the internal Sync mode, the SYNDET output will go to a "one" when the μPD8251 has located the SYNC character in the Receive mode. If double SYNC character (bi-sync) operation has been programmed, SYNDET will go to "one" in the middle of the last bit of the second SYNC character. SYNDET is automatically reset to "zero" upon a Status Read or RESET. In the external SYNC mode, a "zero" to "one" transition on the SYNDET input will cause the μPD8251 to start assembling data character on the next falling edge of $\overline{RxC}$. The length of the SYNDET input should be at least one $\overline{RxC}$ period, but may be removed once the μPD8251 is in SYNC.
16	SYNDET/BD (μPD8251A)	Sync Detect/ Break Detect	The SYNDET/BD pin is used in both Synchronous and Asynchronous modes. When in SYNC mode the features for the SYNDET pin described above apply. When in Asynchronous mode, the Break Detect output will go high when an all zero word of the programmed length is received. This word consists of start bit, data bit, parity bit and one stop bit. Reset only occurs when Rx data returns to a logic one state or upon chip reset. The state of Break Detect can be read as a status bit.

Note ① Since the μPD8251 and μPD8251A will frequently be handling both the reception and transmission for a given link, the Receive and Transmit Baud Rates will be same. $\overline{RxC}$ and $\overline{TxC}$ then require the same frequency and may be tied together and connected to a single clock source or Baud Rate Generator.

Examples

If the Baud Rate equals 110 (Async)
- $\overline{RxC}$ or $\overline{TxC}$ equals 110 Hz (1x)
- $\overline{RxC}$ or $\overline{TxC}$ equals 1.76 KHz (16x)
- $\overline{RxC}$ or $\overline{TxC}$ equals 7.04 KHz (64x)

If the Baud Rate equals 300
- $\overline{RxC}$ or $\overline{TxC}$ equals 300 Hz (1x) A or S
- $\overline{RxC}$ or $\overline{TxC}$ equals 4800 Hz (16x) A only
- $\overline{RxC}$ or $\overline{TxC}$ equals 19.2 KHz (64x) A only

OPERATIONAL DESCRIPTION

A set of control words must be sent to the μPD8251 and μPD8251A to define the desired mode and communications format. The control words will specify the BAUD rate factor (1x, 16x, 64x), character length (5 to 8), number of STOP bits (1, 1-1/2, 2) Asynchronous or Synchronous mode, SYNDET (IN or OUT), parity, etc.

After receiving the control words, the μPD8251 and μPD8251A are ready to communicate. TxRDY is raised to signal the processor that the USART is ready to receive a character for transmission. When the processor writes a character to the USART, TxRDY is automatically reset.

Concurrently, the μPD8251 and μPD8251A may receive serial data; and after receiving an entire character, the RxRDY output is raised to indicate a completed character is ready for the processor. The processor fetch will automatically reset RxRDY.

Note: The μPD8251 and μPD8251A may provide faulty RxRDY for the first read after power-on or for the first read after receive is re-enabled by a command instruction (RxE). A dummy read is recommended to clear faulty RxRDY. But this is not the case for the first read after hardware or software reset after the device operation has once been established.

The μPD8251 and μPD8251A cannot transmit until the TxEN (Transmitter Enable) bit has been set by a Command Instruction and until the $\overline{CTS}$ (Clear to Send) input is a "zero". TxD is held in the "marking" state after Reset awaiting new control words.

USART PROGRAMMING

The USART must be loaded with a group of two to four control words provided by the processor before data reception and transmission can begin. A RESET (internal or external) must immediately proceed the control words which are used to program the complete operational description of the communications interface. If an external RESET is not available, three successive 00 Hex or two successive 80 Hex command instructions (C/$\overline{D}$ = 1) followed by a software reset command instruction (40 Hex) can be used to initialize the μPD8251 and μPD8251A.

There are two control word formats:

1 Mode Instruction
2 Command Instruction

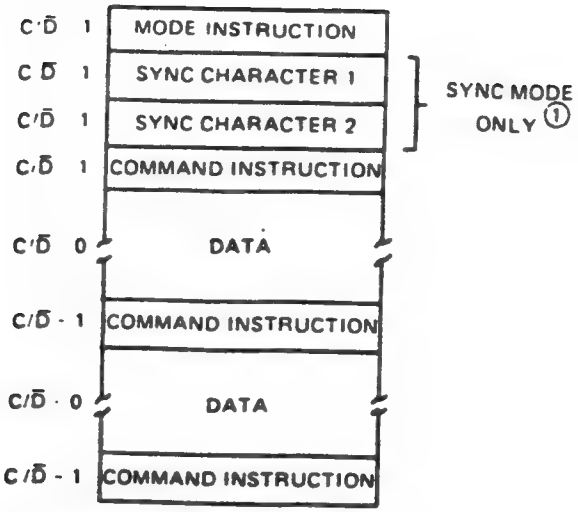
MODE INSTRUCTION

This control word specifies the general characteristics of the interface regarding the Synchronous or Asynchronous mode, BAUD rate factor, character length, parity, and number of stop bits. Once the Mode Instruction has been received, SYNC characters or Command Instructions may be inserted depending on the Mode Instruction content.

COMMAND INSTRUCTION

This control word will be interpreted as a SYNC character definition if immediately preceded by a Mode Instruction which specified a Synchronous format. After the SYNC character(s) are specified or after an Asynchronous Mode Instruction, all subsequent control words will be interpreted as an update to the Command Instruction. Command Instruction updates may occur at any time during the data block. To modify the Mode Instruction, a bit may be set in the Command Instruction which causes an internal Reset which allows a new Mode Instruction to be accepted.

TYPICAL DATA BLOCK



NOTE (1) The second SYNC character is skipped if MODE instruction has programmed the μPD8251 and μPD8251A to single character internal SYNC Mode. Both SYNC characters are skipped if MODE instruction has programmed the μPD8251 and μPD8251A to ASYNC mode.

MODE INSTRUCTION DEFINITION

The μPD8251 and μPD8251A can operate in either Asynchronous or Synchronous communication modes. Understanding how the Mode Instruction controls the functional operation of the USART is easiest when the device is considered to be two separate components (one asynchronous and the other synchronous) which share the same support circuits and package. Although the format definition can be changed at will or "on the fly", the two modes will be explained separately for clarity.

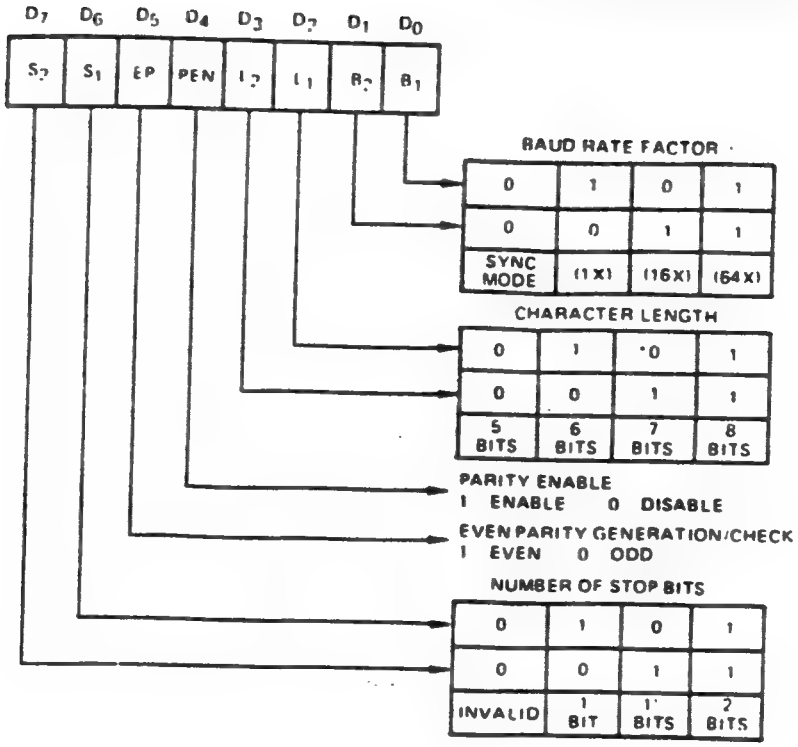
ASYNCHRONOUS TRANSMISSION

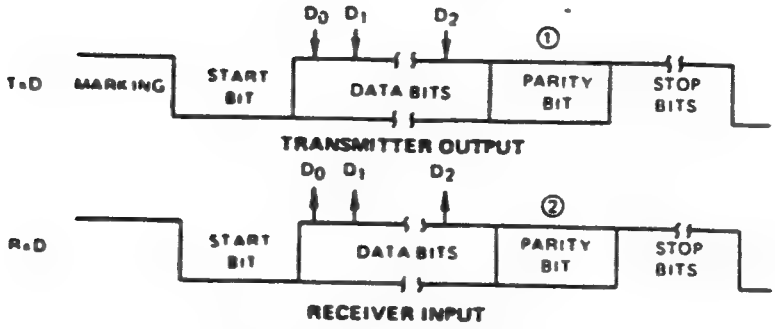
When a data character is written into the μPD8251 and μPD8251A, the USART automatically adds a START bit (low level or "space") and the number of STOP bits (high level or "mark") specified by the Mode Instruction. If Parity has been enabled, an odd or even Parity bit is inserted just before the STOP bit(s), as specified by the Mode Instruction. Then, depending on $\overline{CTS}$ and TxEN, the character may be transmitted as a serial data stream at the TxD output. Data is shifted out by the falling edge of TxC at TxC, TxC/16 or TxC/64, as defined by the Mode Instruction.

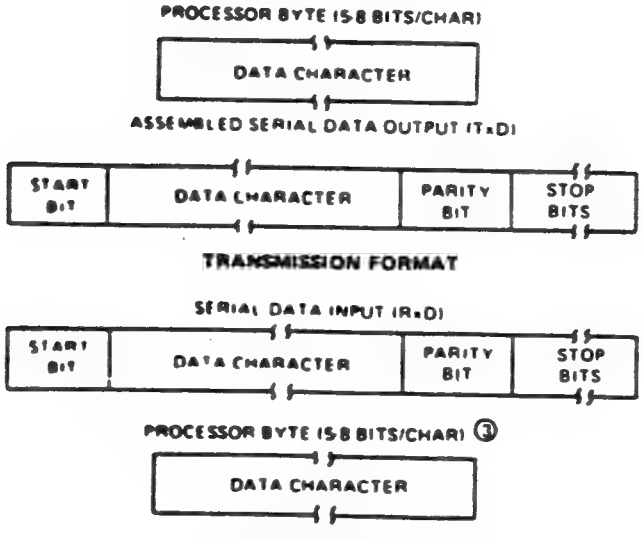
If no data characters have been loaded into the μPD8251 and μPD8251A, or if all available characters have been transmitted, the TxD output remains "high" (marking) in preparation for sending the START bit of the next character provided by the processor. TxD may be forced to send a BREAK (continuously low) by setting the correct bit in the Command Instruction.

ASYNCHRONOUS RECEIVE

The RxD input line is normally held "high" (marking) by the transmitting device. A falling edge at RxD signals the possible beginning of a START bit and a new character. The START bit is checked by testing for a "low" at its nominal center as specified by the BAUD RATE. If a "low" is detected again, it is considered valid, and the bit assembling counter starts counting. The bit counter locates the approximate center of the data, parity (if specified), and STOP bits. The parity error flag (PE) is set, if a parity error occurs. Input bits are sampled at the RxD pin with the rising edge of $\overline{RxC}$. If a high is not detected for the STOP bit, which normally signals the end of an input character, a framing error (FE) will be set. After a valid STOP bit, the input character is loaded into the parallel Data Bus Buffer of the μPD8251 and μPD8251A and the RxRDY signal is raised to indicate to the processor that a character is ready to be fetched. If the processor has failed to fetch the previous character, the new character replaces the old and the overrun flag (OE) is set. All the error flags can be reset by setting a bit in the Command Instruction. Error flag conditions will not stop subsequent USART operation.







Notes: ① Generated by μPD8251/8251A
② Does not appear on the Data Bus
③ If character length is defined as 5, 6, or 7 bits, the unused bits are set to "zero"

SYNCHRONOUS TRANSMISSION

As in Asynchronous transmission, the TxD output remains "high" (marking) until the μPD8251 and μPD8251A receive the first character (usually a SYNC character) from the processor. After a Command Instruction has set TxEN and after Clear to Send ($\overline{CTS}$) goes low, the first character is serially transmitted. Data is shifted out on the falling edge of $\overline{TxC}$ and the same rate as $\overline{TxC}$.

Once transmission has started, Synchronous Mode format requires that the serial data stream at TxD continue at the $\overline{TxC}$ rate or SYNC will be lost. If a data character is not provided by the processor before the μPD8251 and μPD8251A Transmit Buffer becomes empty, the SYNC character(s) loaded directly following the Mode Instruction will be automatically inserted in the TxD data stream. The SYNC character(s) are inserted to fill the line and maintain synchronization until new data characters are available for transmission. If the μPD8251 and μPD8251A become empty, and must send the SYNC character(s), the TxEMPTY output is raised to signal the processor that the Transmitter Buffer is empty and SYNC characters are being transmitted. TxEMPTY is automatically reset by the next character from the processor.

SYNCHRONOUS RECEIVE

In Synchronous Receive, character synchronization can be either external or internal. If the internal SYNC mode has been selected, and the Enter HUNT (EH) bit has been set by a Command Instruction, the receiver goes into the HUNT mode.

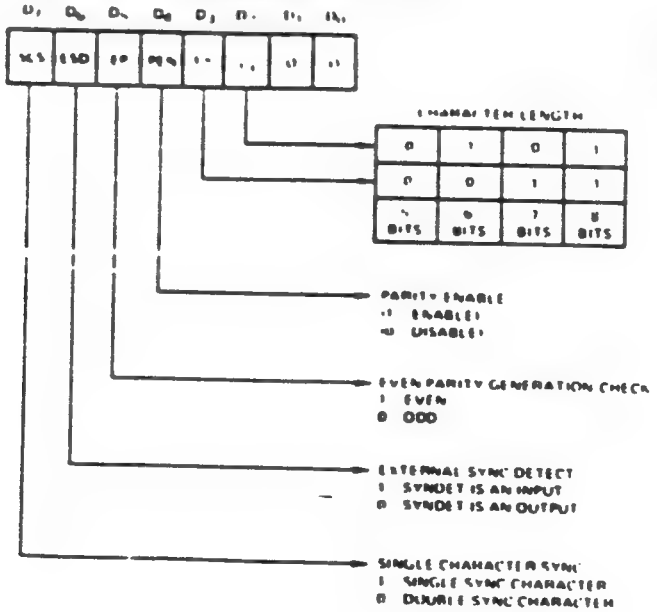
Incoming data on the RxD input is sampled on the rising edge of $\overline{RxC}$, and the Receive Buffer is compared with the first SYNC character after each bit has been loaded until a match is found. If two SYNC characters have been programmed, the next received character is also compared. When the SYNC character(s) programmed have been detected, the μPD8251 and μPD8251A leave the HUNT mode and are in character synchronization. At this time, the SYNDET (output) is set high. SYNDET is automatically reset by a STATUS READ

If external SYNC has been specified in the Mode Instruction, a "one" applied to the SYNDET (input) for at least one $\overline{RxC}$ cycle will synchronize the USART.

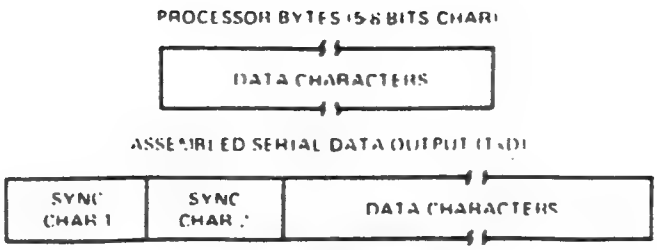
Parity and Overrun Errors are treated the same in the Synchronous as in the Asynchronous Mode. If not in HUNT, parity will continue to be checked even if the receiver is not enabled. Framing errors do not apply in the Synchronous format.

The processor may command the receiver to enter the HUNT mode with a Command Instruction which sets Enter HUNT (EH) if synchronization is lost.

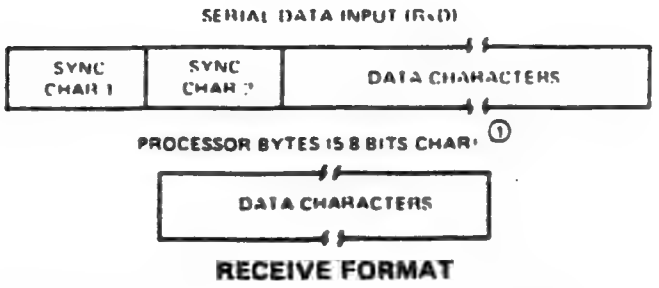
MODE INSTRUCTION FORMAT SYNCHRONOUS MODE



TRANSMIT/RECEIVE FORMAT SYNCHRONOUS MODE



TRANSMIT FORMAT



RECEIVE FORMAT

Note ① If character length is defined as 5, 6 or 7 bits, the unused bits are set to "zero"

COMMAND INSTRUCTION FORMAT

After the functional definition of the μPD8251 and μPD8251A has been specified by the Mode Instruction and the SYNC character(s) have been entered (if in SYNC mode), the USART is ready to receive Command Instructions and begin communication. A Command Instruction is used to control the specific operation of the format selected by the Mode Instruction. Enable Transmit, Enable Receive, Error Reset and Modem Controls are controlled by the Command Instruction.

After the Mode Instruction and the SYNC character(s) (as needed) are loaded, all subsequent "control writes" ($C/\overline{D}$ = 1) will load or overwrite the Command Instruction register. A Reset operation (internal via CMD IR or external via the RESET input) will cause the μPD8251 and μPD8251A to interpret the next "control write", which must immediately follow the reset, as a Mode Instruction.

STATUS READ FORMAT

It is frequently necessary for the processor to examine the status of an active interface device to determine if errors have occurred or if there are other conditions which require a response from the processor. The μPD8251 and μPD8251A have features which allow the processor to read the device status at any time. A data fetch is issued by the processor while holding the $C/\overline{D}$ input "high" to obtain device Status Information. Many of the bits in the status register are copies of external pins. This dual status arrangement allows the μPD8251 and μPD8251A to be used in both Polled and Interrupt driven environments. Status update can have a maximum delay of 16 clock periods in the μPD8251 and 28 clock periods in the μPD8251A.

PARITY ERROR

When a parity error is detected, the PE flag is set. It is cleared by setting the ER bit in a subsequent Command Instruction. PE being set does not inhibit USART operation

OVERRUN ERROR

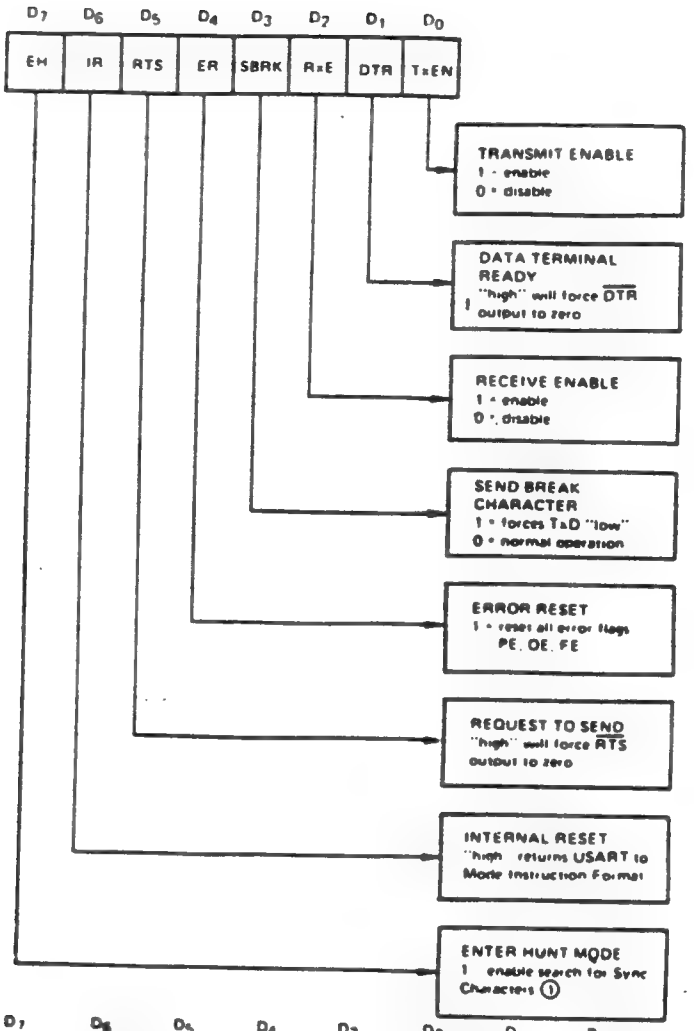
If the processor fails to read a data character before the one following is available, the OE flag is set. It is cleared by setting the ER bit in a subsequent Command Instruction. Although OE being set does not inhibit USART operation, the previously received character is overwritten and lost.

FRAMING ERROR ①

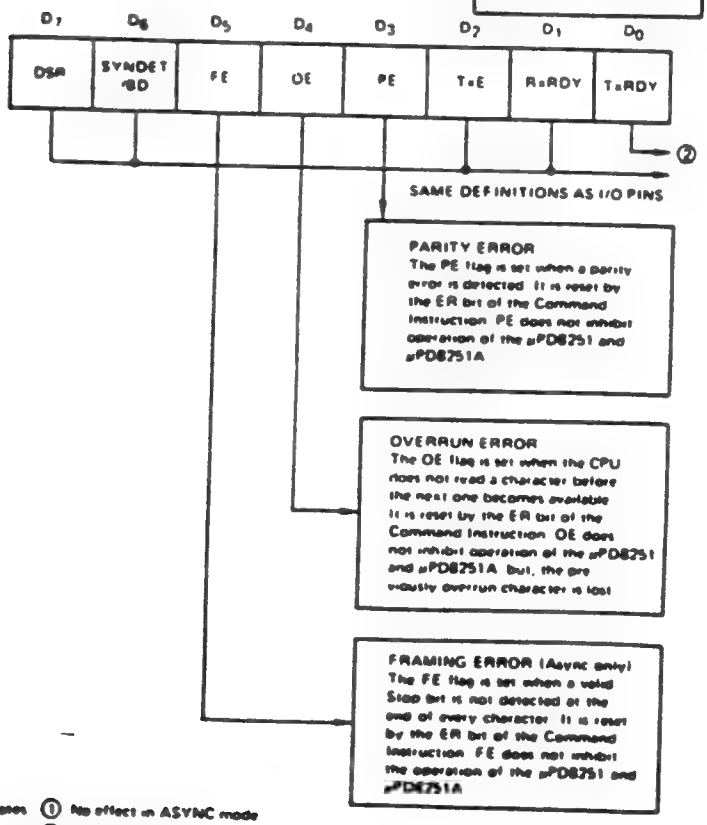
If a valid STOP bit is not detected at the end of a character, the FE flag is set. It is cleared by setting the ER bit in a subsequent Command Instruction. FE being set does not inhibit USART operation.

Note ① ASYNC mode only

COMMAND INSTRUCTION FORMAT



STATUS READ FORMAT



Notes: ① No effect in ASYNC mode

② TxRDY status bit is not totally equivalent to the TxRDY output pin, the relationship is as follows:

TxRDY status bit = DB Buffer Empty

TxRDY (pin 15) = DB Buffer Empty • $\overline{CTS}$ • TxEn

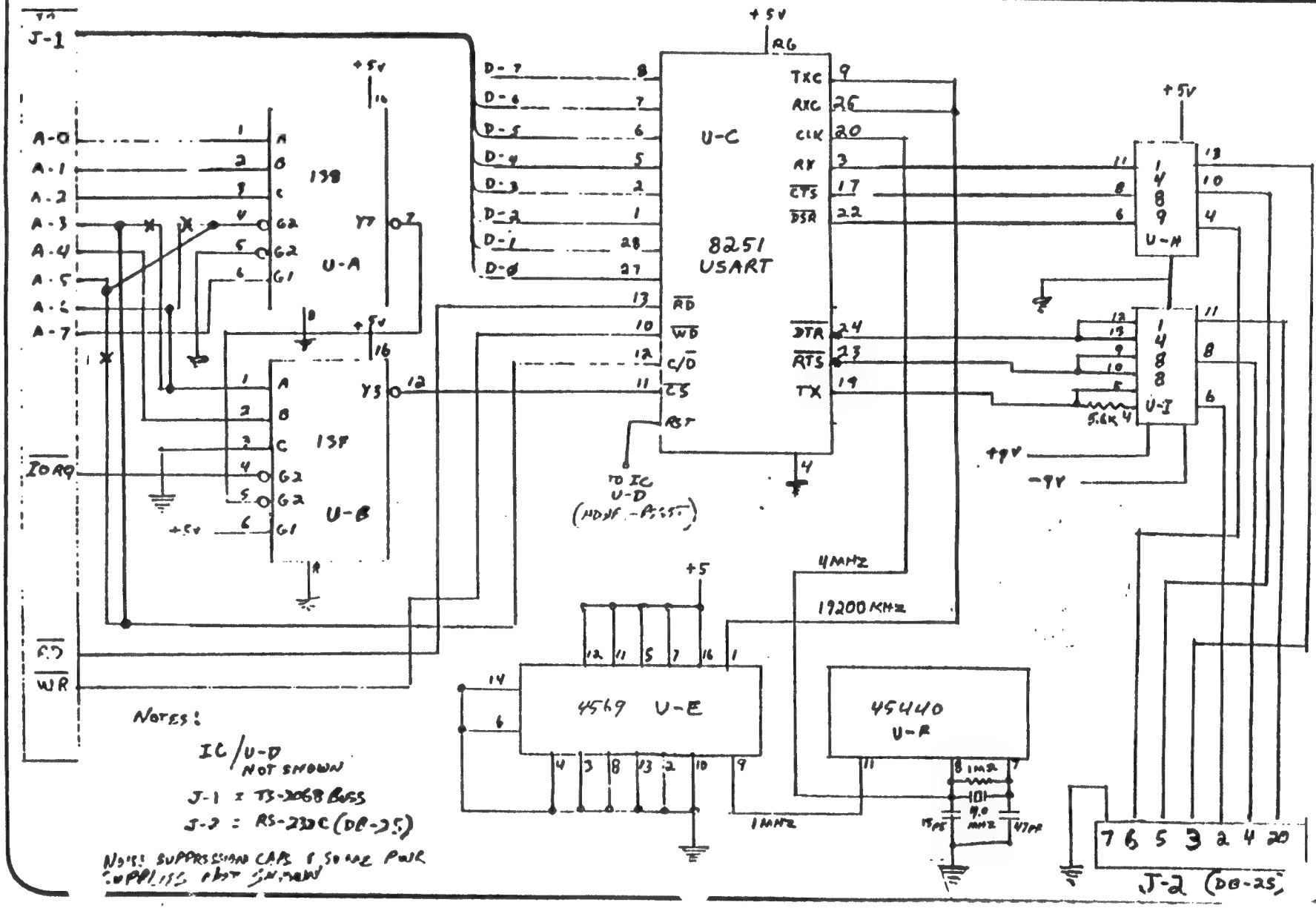
JLO DISK

JLO VERSION PORT ADDRESSES
CONTROL/STATUS = DFH OR 223
DATA IN/OUT = D7H OR 215 } DECIMAL

X = CUT TRACE
— = JUMPERS

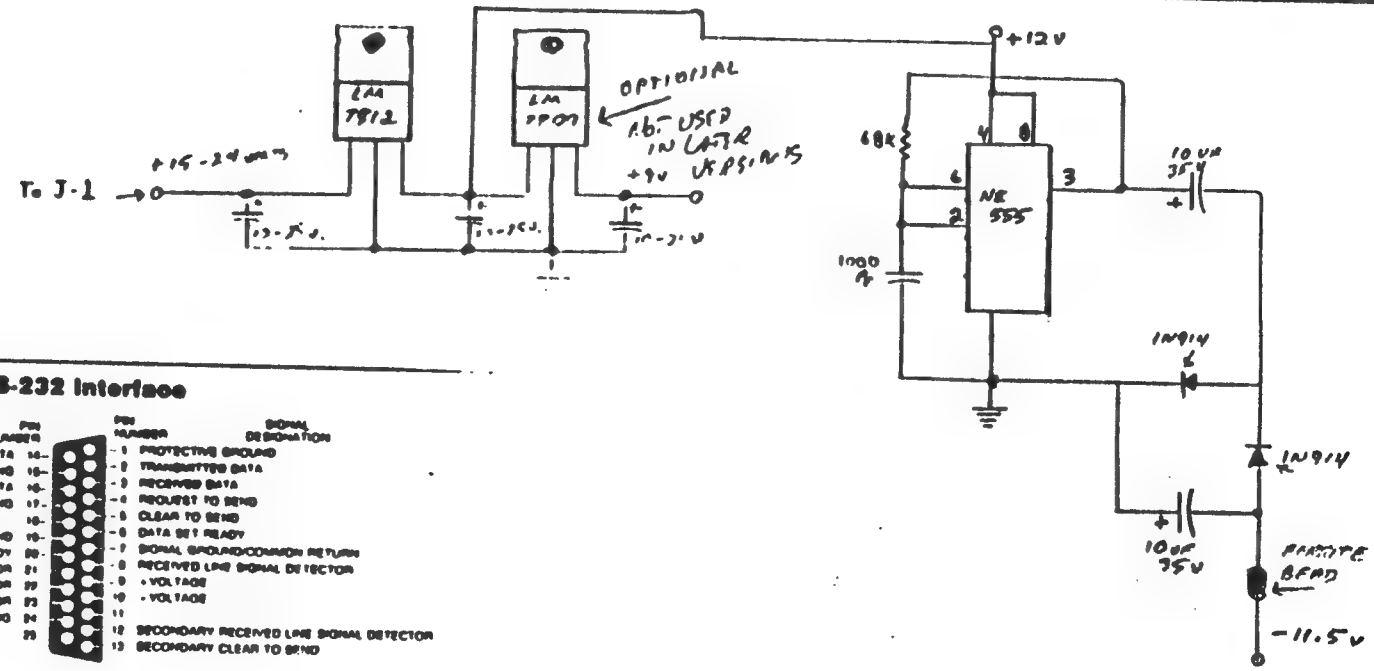
Z-80 I/O Interface Card — Main Logic

NAME: Z-S1/0 REV B
SHEET-1
DEPT: DALE
A602-A16



Z-SI/O Power supply & DB-25 connector 'Pin-Outs'

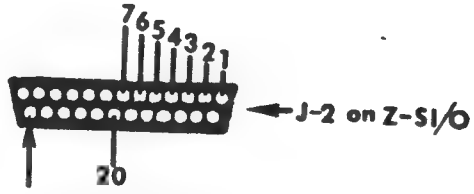
NAME: Z-SI/O Rev B SHEET-2
DEPT: [illegible]
DATE: FEB/07/'86



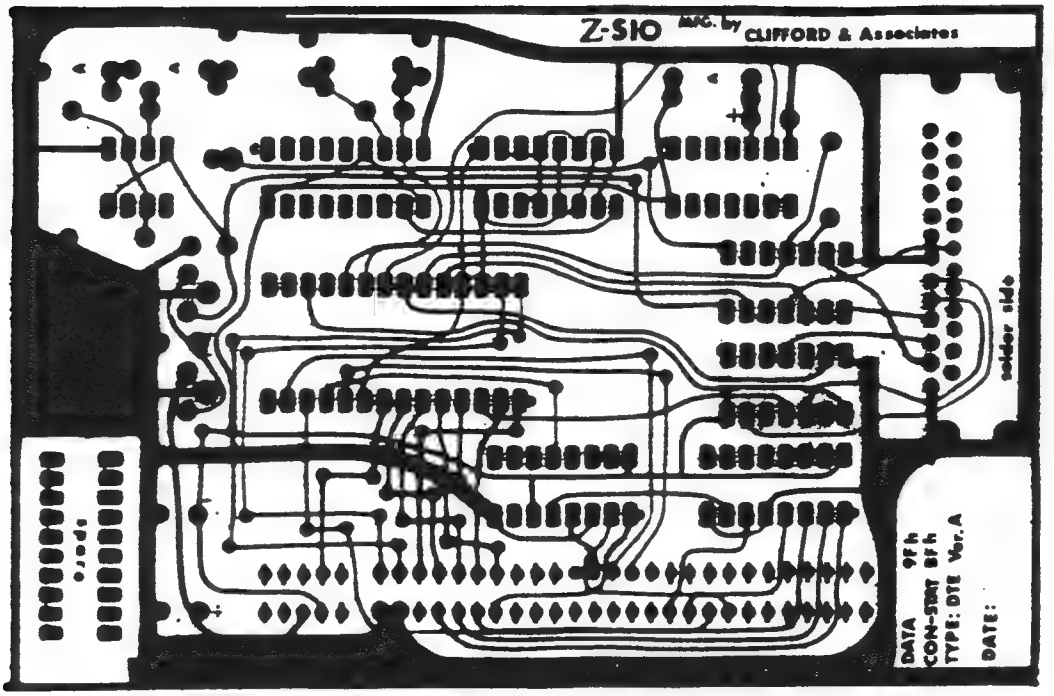
RS-232 Interface

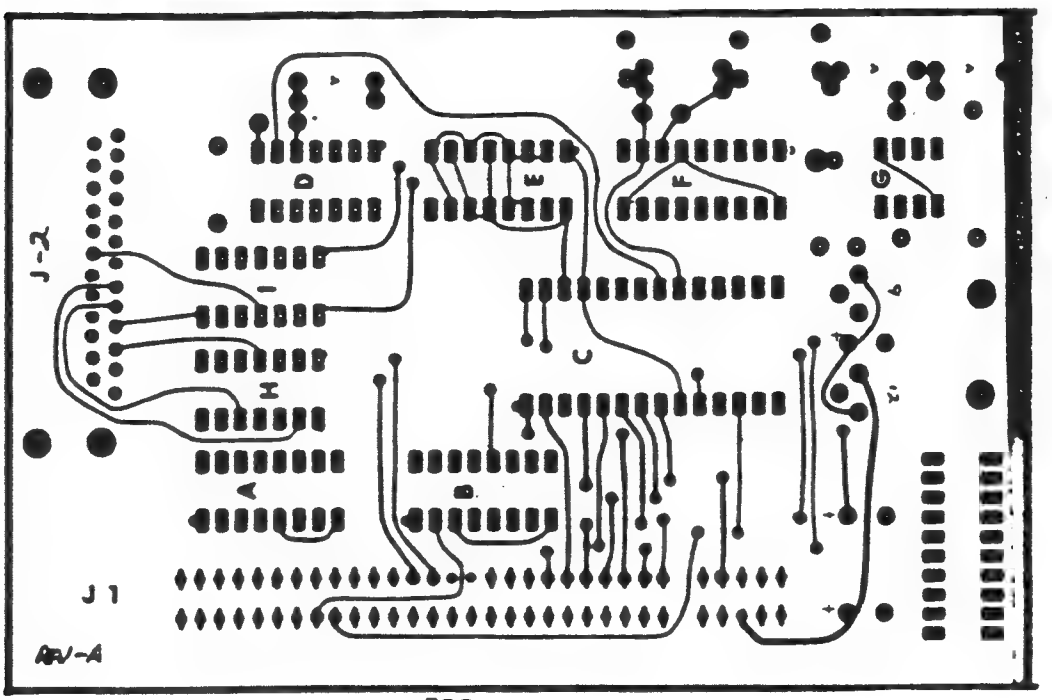
SIGNAL DESIGNATION	PIN NUMBER
SECONDARY TRANSMITTED DATA	14
DCE TRANSMITTER SIGNAL ELEMENT TIMING	15
SECONDARY RECEIVED DATA	16
RECEIVER SIGNAL ELEMENT TIMING	17
	18
SECONDARY REQUEST TO SEND	19
DATA TERMINAL READY	20
SIGNAL QUALITY DETECTOR	21
RING INDICATOR	22
DATA SIGNAL RATE SELECTOR	23
DTE TRANSMITTER SIGNAL ELEMENT TIMING	24
	25

PIN NUMBER	SIGNAL DESIGNATION
1	PROTECTIVE GROUND
2	TRANSMITTED DATA
3	RECEIVED DATA
4	REQUEST TO SEND
5	CLEAR TO SEND
6	DATA SET READY
7	SIGNAL GROUND/COMMON RETURN
8	RECEIVED LINE SIGNAL DETECTOR
9	+VOLTAGE
10	-VOLTAGE
11	
12	SECONDARY RECEIVED LINE SIGNAL DETECTOR
13	SECONDARY CLEAR TO SEND



UN-MARKED Pins are NOT connected
and NOT used on the 'STOCK' Z-SI/O Card.


Z-SIO
MFG. by CLIFFORD & Associates
solder side
DATA 9Fh
CON-STAT 8Fh
TYPE: DTE Ver.A
DATE:


J-2
J 1
REV-A

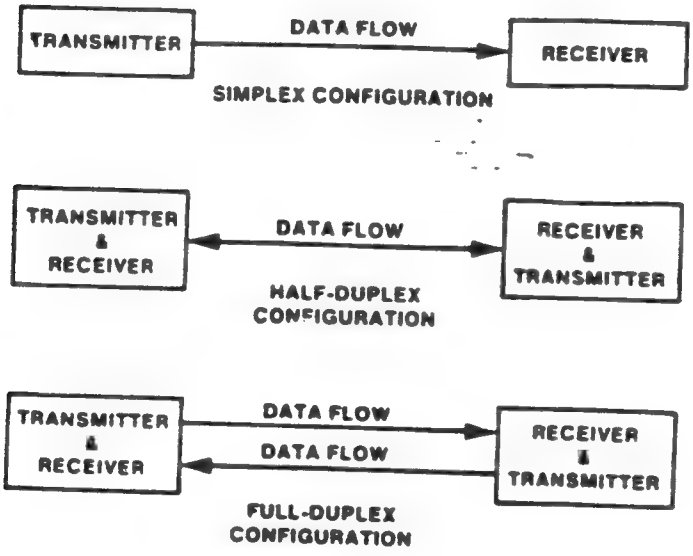
TOP

The following pages have been included for your reference convenience.

Serial, Asynchronous Data Transmission

General

Serial data transmission is characterized by transmitting one data bit at a time between two computing devices. The flow of this data can follow one of three transmission modes: simplex, half- or full-duplex. Simplex only allows data flow in one direction, half-duplex allows data flow in both directions (but not both simultaneously), while full-duplex allows for simultaneous two-way transmissions.



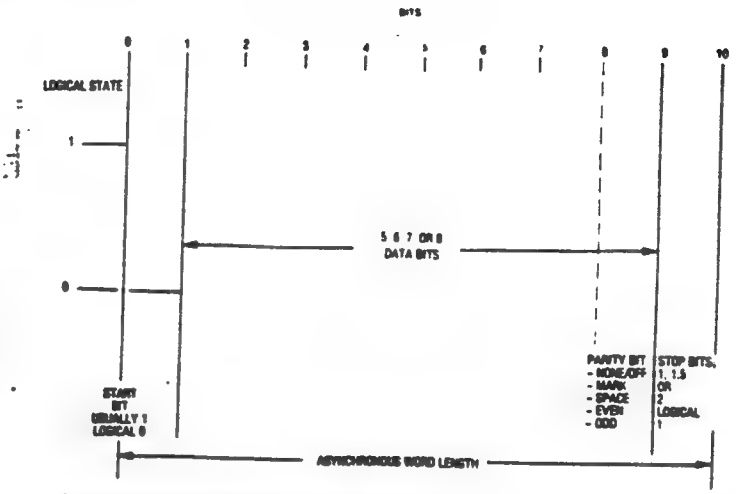
Asynchronous Data Frame

Within the asynchronous data stream, each character of data is transported in a binary bit "frame." This frame has a start bit to signal the beginning of the frame (or character), a stop bit to signal the end of the frame, five to eight data bits to encode the character in binary, and an optional parity bit that encodes the count of 0's and 1's for error detection.

The start bit tells the receiving device to begin looking for following ones and zeros. The start bit is always preceded by a low voltage signal on the data line (mark) which delineates the beginning of a start bit.

The stop bit(s) signals the end of a frame. Stop bits follow the parity bit and may be anywhere from 1-2 bits in length. The slower the speed of transmission, the more stop bits required for end of frame recognition.

Embedded in the heart of every ASCII character frame are 7 or 8 data bits. The collective binary value of the data bits corresponds to a value assigned to an ASCII character (it should be noted that another code such as EBCDIC or baudot would be handled in much the same fashion, with different binary values corresponding to different characters).



Many asynchronous systems append a parity bit following the data bits for error detection. Parity bits trap errors in the following manner: when the transmitting device frames a character, it counts either the number of 0's or 1's in the data bits and appends a parity bit that corresponds to whether or not the count in the data bits was even or odd. The receiving end also counts the data bit 0's or 1's as it receives them and then compares the computation to the parity bit. If an error is detected, a flag can be set and retransmission may be requested. When even parity is chosen, the parity bit is set at 0 if the number of 1's in the data bits is even and it is set at 1 if the number of 1's is odd. Conversely, odd parity sets the parity bit at 1 if the number of 1's in the data bits is even, and it is set at 0 if the number of 1's is odd.

Other parity selections include mark, space or off. Mark parity always sets parity at 1, space parity always sets parity at 0, and off tells the system to ignore the parity bit.

PARITY TYPE	DESCRIPTION
ODD	EIGHTH DATA BIT IS LOGICAL ZERO IF TOTAL NUMBER OF LOGICAL 1'S IN FIRST SEVEN DATA BIT IS ODD
EVEN	EIGHTH DATA BIT IS LOGICAL ZERO IF TOTAL NUMBER OF LOGICAL 1'S IN FIRST SEVEN DATA BIT IS EVEN
MARK	EIGHTH DATA BIT ALWAYS LOGICAL 1 (HIGH/MARK)
SPACE	EIGHTH DATA BIT ALWAYS LOGICALO 0 (LOW SPACE)
NONE/OFF	EIGHTH DATA BIT IGNORED

Asynchronous Link Control

We have defined the asynchronous frame and the directional modes, but we haven't covered the rules for orderly transmission. Each end of the link has to conform to certain operating parameters in order for flow to occur without loss of data.

The first parameter that must be established is the rate of transmission. Serial data transmission is measured in bits per second (bps). Typical asynchronous serial baud rates are 110, 300, 600, 1200 and 2400 bps. To have an interactive session between two computing devices, both of the devices must either be transmitting and receiving data at the same baud rate, or there must be an intermediate memory device, called a buffer, that accommodates the differences in speed.

In addition to transmit and receive buffers, most serial asynchronous links use a flow control system to handle data transmission. A common asynchronous serial flow control is the X-ON/X-OFF protocol. When a receive buffer reaches its memory capacity, the receiving device sends an ASCII DC3 (X-OFF) to the transmitting device telling it to stop sending data. When the receive buffer has sufficiently unloaded, it transmits an ASCII DC1 (X-ON) back to the transmitting device, telling it to continue transmission.

Other flow controls commonly used in asynchronous communications are lead control and the ENQ/ACK protocol. A lead control protocol controls data flow by alternately raising and lowering the voltage on a lead of the RS-232 interface. DTR (pin 20) and CTS (pin 5) are typically used. When the pin is high, data flow is enabled; when it drops low, data flow is stopped.

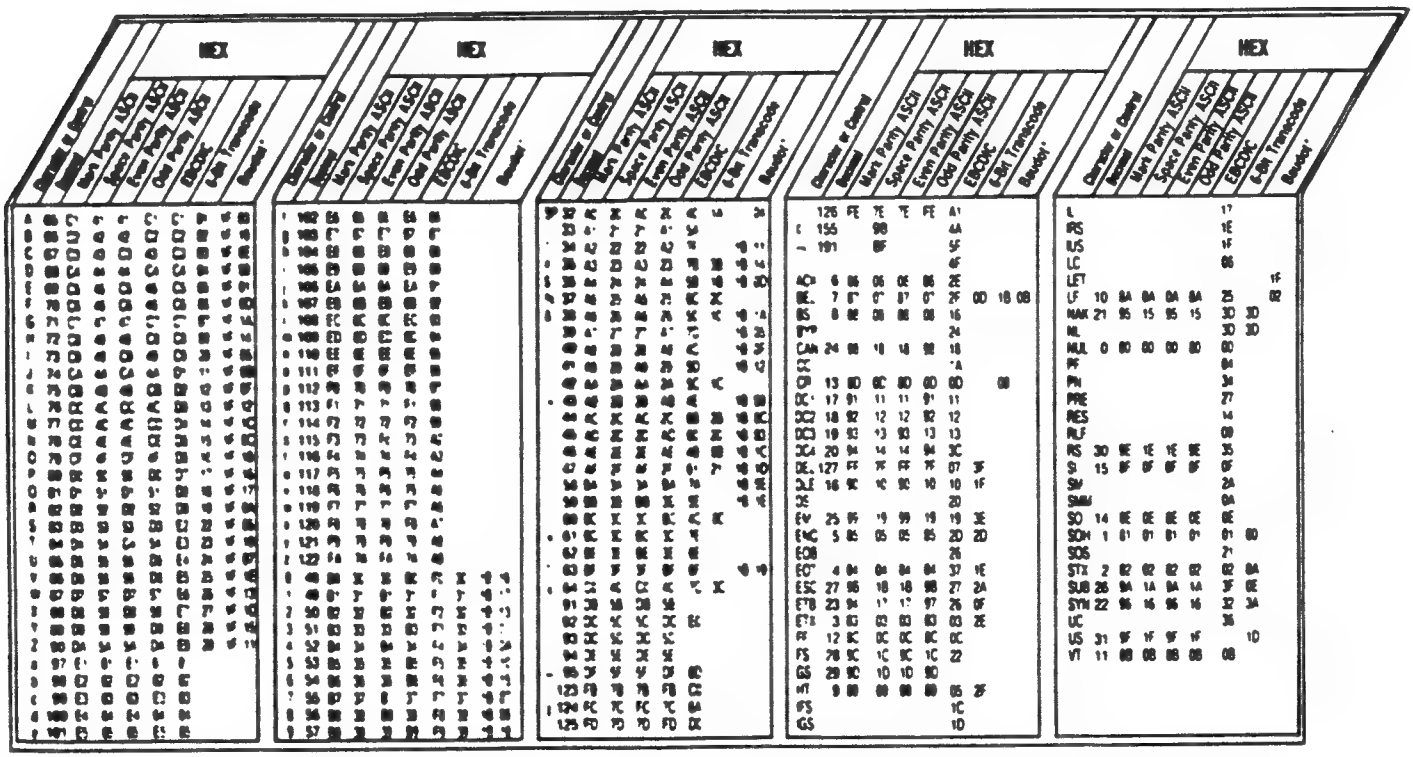
ENQ/ACK is a block-oriented protocol, i.e. a protocol that sends a fixed block-size of characters everytime it transmits. Typically, the transmitting device sends an ENQ character and waits for an ACK character from the receiving device before beginning transmission. Once the ACK character is received, the entire block is transmitted. When the block transmission is complete, the transmitting device once again begins polling the receiving device with ENQs.

Binary Data Encoding

Computer data streams are nothing more than a sequence of electrical highs and lows, (ones and zeros), yet the ability to differentiate ones and zeros allows us to code virtually anything. A simple eight-bit sequence of ones and zeros can produce any number from 0 - 255. Computer scientists assigned these 256 binary values to a corresponding character set. The character set includes all of the letters and numbers, punctuation, and a variety of control characters. As a result, every time a computer receives a binary 01000010 (decimal 66, hex 42), it interprets it as the character "B" or, similarly, every time a computer receives a binary 01001001 (decimal 73, hex 49), it interprets it as a "1." There are more than one of these code sets, but the most common is ASCII or American Standard Code for Information Interchange. ASCII was developed by the American National Standards Institute and has attained a nearly universal acceptance. EBCDIC or Extended Binary Coded Decimal Interchange Code has been used widely in many IBM applications and baudot and transcode are leftovers from ticker tape and Telex® applications.

Because eight-bit binary values are so hard to read, we usually look at the numeric representation of a character in hexadecimal (base 16) or decimal (base 10). Now, remember that binary numbers are comprised of zeros and ones, and decimal numbers are comprised of numbers zero through nine; therefore, to construct a hexadecimal (base 16) number, we need to have numbers for 0 - 15. To do this we add A, B, C, D, E and F to the numbers 0 - 9, i.e. "A" corresponds to decimal 10, B to 11, C to 12, D to 13, E to 14, and F to 15. Then the number 16 becomes 10 in hexadecimal (note that the 10's column in decimal has become the 16's column in hexadecimal).

We have included the chart below to give you a complete cross reference between character code sets and their decimal and hexadecimal representations. Note that the ASCII hex value changes depending upon the parity selected. This is because the parity bit is appended to the end of the binary representation of the character and therefore changes the resulting binary and hexadecimal numbers.



A few words about Synchronous Data Transmission

Synchronous Data Transmission is based on the concept of using special synchronous characters to synchronize the transmitting and receiving elements of the link; this permits transmissions to occur without the overhead of start and stop bits as in asynchronous communication. Synchronous protocols fall into two major categories: 1) character- or byte-oriented protocols that specify a definite character length, and 2) bit-oriented protocols that do not specify character boundaries.

Bisynchronous (BSC), short for Binary Synchronous Communication, is one of the most common character-oriented protocols. Bisynchronous uses a set of special characters to define the structure of the data transmission frame. At the start of a block of data are a couple of PAD and SYN characters which signal the start of a frame and allow the receiving station to synchronize with the transmitting station clock. A variety of different bisynchronous transmission frames can be created using the available character control set. Below we show the list of characters and a typical bisynchronous frame:

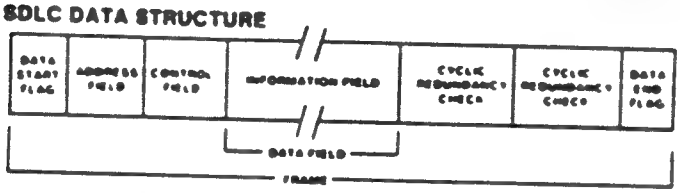
BISYNC DATA STRUCTURE

BISYNC CONTROL CHARACTERS

BISYNC CHARACTER	HEX VALUE	CHARACTER DESCRIPTION
SYN	32	Synchronous idle
PAD	55	Start of frame pad
PAD	FF	End of frame pad
DLE	10	Data line escape
ENQ	2D	Enquiry
SOH	01	Start of heading
STX	02	Start of text
ITB	1F	End of intermediate block
ETB	26	End of transmission (block)
ETX	03	End of text

SDLC, short for Synchronous Data Link Control, typifies the second type of synchronous protocol — the bit-oriented protocol. Instead of using a control character set as does bisynchronous, SDLC uses a variety of bit patterns to flag the beginning and end of an SDLC frame. Other bit patterns are used for the address, control and packet header fields which route the frame through a network to its destination. Below we show a typical SDLC transmission frame.

SDLC DATA STRUCTURE



Please note that a complete description of synchronous data transmission is out of the scope of this limited space. We recommend the following for further reference:

IBM® Publications:

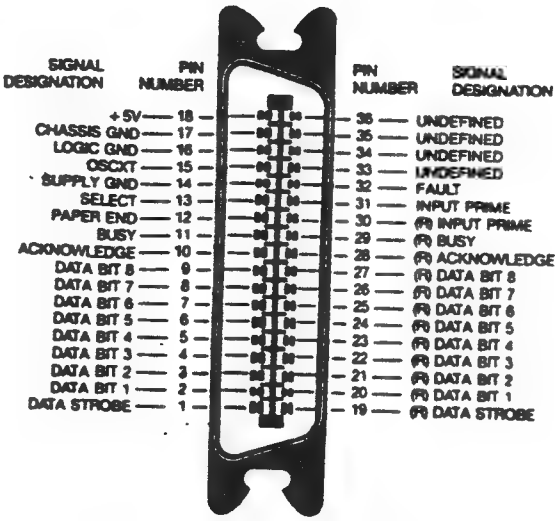
Binary Synchronous Communications —
IBM Order No. 6A27-3004-4

Synchronous Data Link Control —
IBM Order No. 6A27-3093-2

Centronics® Parallel Interface

The Centronics Parallel Interface is a 36-pin, byte-wide interface that has become a widely accepted standard for computer to printer communications. The interface has eight lines which carry their respective binary bits in parallel. The transmission of these data bits is controlled by the computer supplied STROBE pulse. Handshaking (flow control) is achieved by asserting or deasserting either the ACKNLG or BUSY leads or both. All Centronics Parallel logic levels are TTL.

We have included a diagram of the Centronics connector and a chart that covers the most commonly used leads for your reference convenience.



(R) INDICATES SIGNAL GROUND RETURN

Function and Direction of Most Commonly Used Centronics Interface Leads

SIGNAL PIN NO.	RETURN PIN NO.	SIGNAL	DIRECTION (with ref. to printer)	DESCRIPTION
1	19	STROBE	In	STROBE pulse (negative going) enables reading data.
2	20	DATA 1	In	1st to 8th bits of parallel data. Each signal is at "HIGH" level when data is logical "1" and "LOW" when logical "0".
3	21	DATA 2	In	
4	22	DATA 3	In	
5	23	DATA 4	In	
6	24	DATA 5	In	
7	25	DATA 6	In	
8	26	DATA 7	In	
9	27	DATA 8	In	
10	28	ACKNLG	Out	"LOW" indicates that data has been received and that the printer is ready to accept other data.
11	29	BUSY	Out	"HIGH" indicates that the printer cannot receive data.

NOTE: Pins 12, 13, 14, 15, 18, 31, 32, 34, 35 and 36 vary in function depending upon implementation; they are commonly used for printer auxiliary controls, and error handling and indication.

Pins 16 and 17 are commonly used for logic ground and chassis ground, respectively.

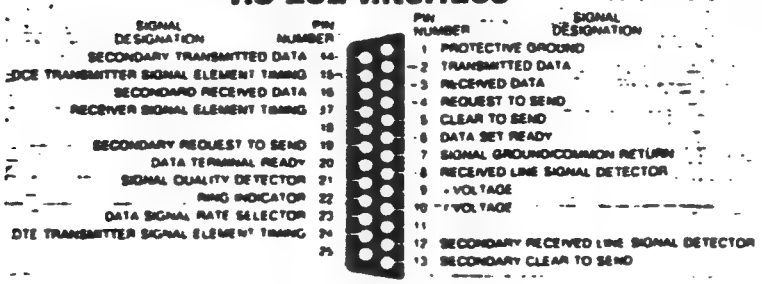
RS-232 and RS-449 Interfaces

RS-232 is an EIA standard, applicable to the 25-pin interconnection of Data Terminal Equipment (DTE) and Data Communication Equipment (DCE) employing serial binary data interchange.

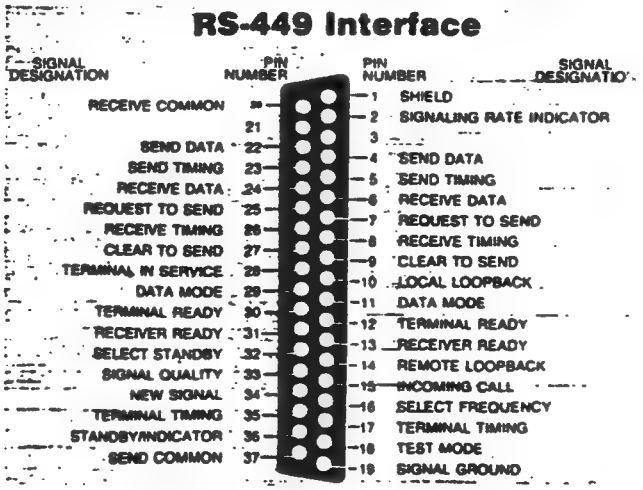
RS-449 is an EIA standard, applicable to the 37-pin and 9-pin interconnection of Data Terminal Equipment (DTE) and Data Communication Equipment (DCE) employing serial binary data interchange.

Both of these standards have been widely accepted in computer and data communications technologies. We have included the following diagrams and chart for your reference convenience.

RS-232 Interface



RS-449 Interface



Pinout Table For EIA RS-449, EIA RS-232C/CCITT V.24

RS-449 Interface					RS-232 Interface				Signal Type & Direction							
	37 PIN								GND	DATA		CONTROL		TIMING	
9 PIN ASSY.	A	B	RS449 CIRCUIT	RS449 DESCRIPTION	25 PIN	EIA-RS232C CIRCUIT	CCITT-V.24 CIRCUIT	RS232 DESCRIPTION		From DCE	To DCE	From DCE	To DCE	From DCE	To DCE
1	1			Shield	1	AA	101	Protective Ground	X						
5	19		SG	Signal Ground	7	AB	102	Signal Ground/Common Return	X						
9	37		SC	Send Common			102a	DTE Common	X						
8	20		RC	Receive Common			102b	DCE Common	X						
	4	22	SD	Send Data	2	BA	103	Transmitted Data			X				
	6	24	RD	Receive Data	3	BB	104	Received Data		X					
	7	25	RS	Request to Send	4	CA	105	Request to Send					X		
	9	27	CS	Clear to Send	5	CB	106	Clear to Send				X			
	11	29	DM	Data Mode	6	CC	107	Data Set Ready				X			
	12	30	TR	Terminal Ready	20	CD	108.2	Data Terminal Ready					X		
	15		IC	Incoming Call	22	CE	125	Ring Indicator				X			
	13	31	RR	Receiver Ready	8	CF	109	Received Line Signal Detector				X			
	33		SQ	Signal Quality	21	CG	110	Signal Quality Detector				X			
	16		SR	Signaling Rate Selector	23	CH	111	Data Signal Rate Selector (DTE)					X		
	2		SI	Signaling Rate Indicator	23	CI	112	Data Signal Rate Selector (DCE)				X			
	17	35	TT	Terminal Timing	24	DA	113	Transmitter Signal Element Timing (DTE)							X
	5	23	ST	Send Timing	15	DB	114	Transmitter Signal Element Timing (DCE)						X	
	8	26	RT	Receive Timing	17	DD	115	Receiver Signal Element Timing (DCE)						X	
3			SSD	Secondary Send Data	14	SBA	118	Secondary Transmitted Data			X				
4			SRD	Secondary Receive Data	16	SBB	119	Secondary Received Data		X					
7			SRS	Secondary Request to Send	19	SCA	120	Secondary Request to Send					X		
8			SCS	Secondary Clear to Send	13	SCB	121	Secondary Clear to Send				X			
2			SRR	Secondary Receiver Ready	12	SCF	122	Secondary Received Line Signal Detector				X			
	10		LL	Local Loopback			141	Local Loopback					X		
	14		RL	Remote Loopback			140	Remote Loopback					X		
	18		TM	Test Mode			142	Test Indicator				X			
	32		SS	Select Standby			116	Select Standby					X		
	36		SB	Standby Indicator			117	Standby Indicator				X			
	16		SF	Select Frequency			126	Select Transmit Frequency				X			
	28		IS	Terminal in Service									X		
	34		NS	New Signal									X		

NOTE: References DB25 (25-pin) connector that is commonly used for RS-232 and V.24. Pins 9 and 10 are reserved for data set testing. Pins 11, 18 and 25 are undefined. Pins 3 and 21 of RS-449 interface connector are undefined. Lead 23 of RS-232 connector may be defined as CH or CI. 37-pin designation B equals return.

"Z-PRINT"
Z-SI/O SERIAL INTERFACE CARD
PRINTER DRIVER SOFTWARE

Below is a BASIC listing of a machine code routine to serve as a simple 'no-frills' printer driver configured for the Z-SIO card. After typed in SAVE it to tape. When LOADed the machine code (contained in the DATA statement) is POKED in to place by lines 10, 20, 30 and then the parameters for the UART on the Z-SI/O are set by lines 40 thru 90. Lines 200 to 230 and 310 to 330 POKE the machine code depending on wether you are using the 'stock' 2068 or a 'SPECCY' emulated machine. This MC driver uses an 'off-set' to compare key-words against the ROM based TOKEN to ASCII look-up table and is used with the LLIST function. Also the machine variable 'CHANS' is poked with the proper address.

PORT ADDRESSES for the DATA and STATUS ports.
As stated above lines 40 thru 90 'program' the UART. The Z-SI/O card is available in two configurations ('STOCK' and 'OLIGER DISK' versions) each using different I/O port addresses. The listing AS SHOWN is for the 'STOCK' Z-SI/O addresses (CON/STAT port = 191 and DATA = 159). The 'JLO DISK' version uses port 223 for CONT/STAT and port 215 for DATA. If you are using the OLIGER version be sure to change the '191' in lines 40 thru 90 to '223' and after your MC has been LOADED you must POKE the following:

POKE 65033,223 POKE 65070,215 POKE 65074,215

Now your driver will work with the 'JLO-DISK' version of the Z-SI/O card. You can SAVE the driver MC with SAVE "name"CODE 64964,299
Your BASIC loader should contain lines 40 to 90 to INIT the UART. This only has to be done ONCE or when you want to change BAUD rates or if you have shut-off the computer. CLEAR and NEW does not affect the UART settings but you may want to set RAMTOP below 64964 to protect your driver from being over-written by another program.

FEATURES

TWO PRINTING MODES...

The driver has two modes of operation. TOKEN PRINTING and NON TOKEN PRINTING. When loaded the driver 'defaults' to TOKEN PRINTING which allows the LLISTing of BASIC listings. You may turn this feature OFF by POKEing address 23296 with ANY number except '0' and you may turn it back ON by re-POKEing 23296 with '0'. The next page has more info about what each mode will do.

TOKEN PRINTING (default) MODE...

After LOADed and initalized ANYTHING normally sent to the TIMEX 2040 printer via the LPRINT or LLIST commands will be directed to the printer attached to the Z-SI/O as printable ASCII codes. If you use the LLIST command all BASIC TOKEN codes are converted and expanded to ASCII and sent to the printer.
An automatic LINE FEED (ASCII 0Ah) is SENT at the end of a line or after a CARRIAGE RETURN (ASCII 0Dh).

NON TOKEN PRINTING MODE....

Same as above except BASIC TOKENS are NOT expanded to ASCII thus allowing ANY code between 0 to 255 to be sent to the printer. For example to send a LINE FEED to the printer from BASIC use: LPRINT CHR$(10) or to send an ESCAPE command sequence use: LPRINT CHR$(27);CHR$(x) where 'x' is the ASCII code for the control character. Also NON-TOKEN mode allows codes above 128 to be sent to the printer. This is handy for printers that support 'bit-mapped' graphics.

BREAK (CAP-SHIFT/BREAK)

You may STOP sending data to the printer at ANY time by pressing the BREAK and CAP-SHIFT keys. This will pop you back to BASIC and you need not worry about 'locking-up' the computer if something goes wrong like a paper jamb etc. NOTE: When you hit BREAK the routine will send a CARRIAGE RETURN to allow the printer to finish printing the contents of it's buffer.

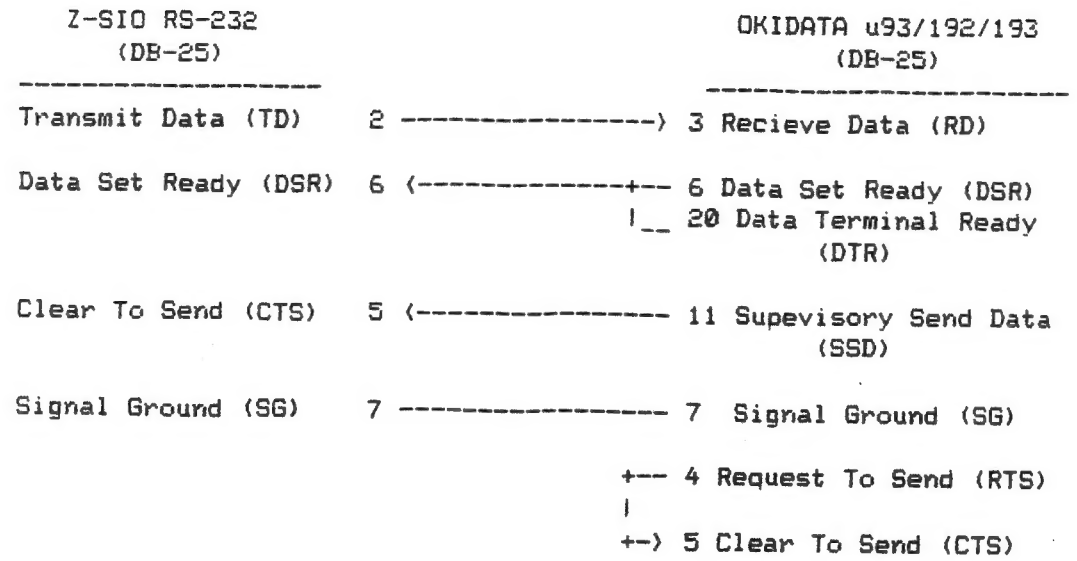
HANDSHAKING...

'HANDSHAKING' is a buzz-word used to describe the type of control that is employed by the computer and the device it is 'talking' to. Most serial printers allow for a choice between two popular types. 'X-ON/X-OFF' and 'SSD' (Supervisory Send Data). The 'Z-PRINT' driver supports SSD only. For a detailed discussion on the protocols that are used by different systems and how they work please see the TECHNICAL REFERENCE section starting on page 23 in the Z-SI/O Serial Inteface Manual. We will only be concerned with the protocol that 'Z-PRINT' supports here.

CONNECTIONS...

First, consult your PRINTER MANUAL and set it up for an 8 BIT data word, ZERO parity and 1 STOP. Then consult the section as to how to set the printer for use with 'SSD' or "PRINTER BUSY" protocol. The 'SSD' line from the printer should be tied to the 'CTS' (pin 5) of the Z-SI/O RS-232 connector.

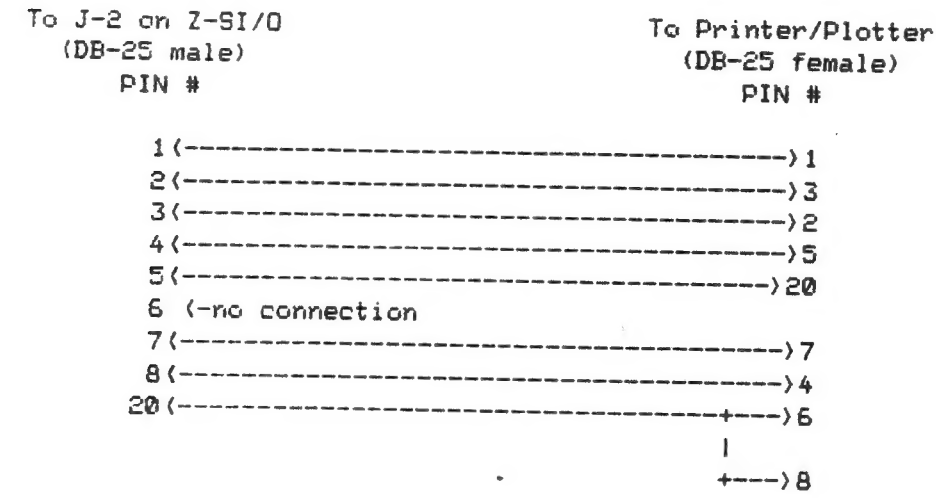
For example if you have an OKIDATA printer with the 'SUPER SPEED RS-232c' card installed you would wire your printer interface cable as shown on the following page....



RE-SOURCES....

If you would like more info on the 'skeleton' of this driver see issue #7 of YOUR SPECTRUM magazine. Z-SI/O settings are done from BASIC 'OUT' statements. Also see the section on programming in the Z-SI/O Manual (Page 4) and the 8251 UART Data Sheet (Page 9 to 19a).

This driver has been tested with the Z-SI/O and the OKIDATA Microline 93s printer as well as the APPLE-410 Color Plotter. To use the Z-SI/O with a serial printer or the Apple plotter you must wire a 'configuration' connector or "gender-changer" as shown below. This adapter will configure the Z-SI/O as a printer port (the 2068 can be thought of as a 'modem' rather than a 'terminal'). This can be used to connect the 2068 directly to a second computers modem port too. Then by using suitable software you could transfer files between the two machines.



A General-Purpose "GENDER-CHANGER" for the Z-SI/O

"Z-PRINT"

A 'universal' serial printer/plotter driver adapted from "YOUR SPECTRUM" magazine, Sept. 1985. It supports LPRINT, LLIST and CHR$ (n) control codes. It was written to work with the CLIFFORD & Associates Z-SI/O RS-232c serial interface card by Michael J. Di Rienzo, ANAHIEM, CA Oct. 1986 This software may be modified to suit your needs.

Type in the folowing listing and SAVE to tape.

```
  1 DATA 254,032,048,011,000,000,024,059,071,058,000,091,183,
200,120,254,165,056,048,071,058,000,091,183,120,032,040,214,165,
038,000,111,041,017,058,254,025,094,035,086,033,150,000,025,062,
032,205,007,254,126,203,127,032,006,205,007,254,035,024,245,203,
191,205,007,254,062,032,245,219,191,203,071,032,005,205,023,254,
024,245,241,024,027,201,062,127,219,254,031,216,062,254,219,254,
031,216,033,016,039,043,124,181,032,251,062,013,211,159,207,012,
211,159,254,013,192,062,010,024,205,000,000,003,000,009,000,011,
000,013,000,018,000,025,000,029,000,031,000,034,000,038,000,042,
000,045,000,048,000,051,000,054,000,057,000,060,000,063,000,066,
000,068,000,071,000,074,000,077,000,080,000,083,000,087,000,089,
000,092,000,096,000,100,000,103,000,106,000,108,000,111,000,113,
000,115,000,117,000,121,000,125,000,127,000,131,000,137,000,140,
000,146,000,150,000,155,000,161,000,168,000,173,000,179,000,183,
000,189,000,192,000,197,000,202,000,208,000,215,000,219,000,222,
000,228,000,233,000,237,000,241,000,245,000,252,000,255,000,005,
001,013,001,016,001,019,001,022,001,027,001,033,001,038,001,042,
001,046,001,049,001,054,001,058,001,062,001,067,001,071,001,074,
001,078,001,087,001,089,001,092,001,096,001,101,001,107,001
   10 FOR N=64964 TO 65263
   20 READ A: POKE N,A
   30 NEXT N
   40 FOR N=1 TO 3
   50 OUT port,0
   60 NEXT N
   70 OUT port,64
   80 OUT port,110 (-111 FOR 300 BAUD
   90 OUT port,17
  100 STOP
  200 REM FOR TS-2068 ADD--
  210 POKE 26703,196
  220 POKE 26704,253
  230 POKE 65005,153
  240 STOP
  300 REM FOR SPECTRUM ADD--
  310 POKE 23749,196
  320 POKE 23750,253
  330 POKE 65005,150
  340 STOP
 9999 SAVE "Z-PRINT"
```

NOTES:

For 'STOCK' Z-SI/0
port = 191

For 'JLO-DISK' Z-SI/O
port = 223

Also POKE 65033,223
POKE 65070,215
POKE 65074,215

MORE RESOURCES

The driver just described is the 'better' of two that are available for the Z-SI/O. As time passes I am sure others will be created by the many ingenious programmers and users and hopefully will be realesed through the various magazines and user support groups. The 'second' driver I mentioned that is available is the "UP-ARROW" driver from the 'PRO-FILE 2068' manual by Thomas B. Woods (an assembly listing is included). This driver does NOT support LLIST but is unique due to the 'up-arrow' feature. You may find this routine useful in your own project. If you prefer to NOT type-in the 'Z-PRINT' or do not have an assembler like HOT-Z, ZUES or the Picturesque EDITOR/ASSEMBLER to assemble the 'up-arrow' driver there are two ways to obtain them ready-to-run. They are available for DOWNLOAD on the TIME-=(X)=-CHANGE RBBS-RCP/M at (213) 329-3922 If you use MTERM-II in HEX mode the 'Z-PRINT' driver may be found in SECTION TS-2068 as 'Z-DRIVE.TS' and 'Z-DRIVE.DOC' (this file). If using Casby's Loader-V XMODEM or SPECTERM-64 you will find both drivers and their associated doc and assembly source listings and BASIC loaders on SECTION SPECTERM in a library file called 'Z-TOYS.LBR' and use the LUX utility to access and down-load the files. Or if you prefer send $5.00 check/M.O. to DAVE CLIFFORD 13910 Halldale Ave. Gardena, California 90249 and I will send the 'Z-PRINT' printer driver plus a 'compiled' version of the example 'DUMB-TERMINAL' routine discribed on page #6 in the Z-SI/O manual on a cassette with a copy of these docs post-paid. (ask for the "TOY-BOX" cassette.) I assume most Z-SI/O users will have the SPECTERM-64 Terminal Software and if you do, a visit to the TIME-=(X)=-CHANGE may prove most rewarding when hunting for software for the Z-SI/O and SPECTERM-64 that has been contributed by other users.

FOOD FOR THOUGHT

Because of the many different ways that various printers handle 'BIT-MAPPED' graphics and the fact that the 'Z-PRINT' and 'UP-ARROW' drivers do not support the COPY command all is not lost if you are 'game' for a challenge. In YOUR SPECTRUM magazine starting with ISSUE #4 (June 1985) there is an article by Andrew Pennell called 'DUMPS of DISTINCTION' in which he lays the ground work for a hi-rez screen dump routine for several of the more popular parallel printer drivers in the U.K. for the 'SPECCY' to drive the Epson FX-80 and "look-alikes". The routines even have a 'grey-scale' for color. Making any parallel driver work with a serial I/F is not too difficult a task. (Both 'Z-PRINT' and 'UP-ARROW' were parallel drivers in their previous forms.)

Amateur Radio operators may find SPECTERM-64 and the Z-SI/O Interface Card an ideal combination for use with a TNC for 'PACKET' radio communications on the SPECTRUM 'emulated' T/S2068. You could use the 'Z-PRINT' driver to 'dump' text or even binary files to another computer fitted with an RS-232 port (say a QL, CP/M or even an IBM machine).

Z-SI/O SERIAL INTERFACE CARD
"DUMBTERM" (example terminal routines)

On side 'B' of the "TOY-BOX" cassette there are 3 versions of the 'EXAMPLE' dumb-terminal routine that can be found on page 6 of the Z-SI/O TECHNICAL/PROGRAMMING MANUAL. The routine as written in BASIC may be 'OK' for use in a 'local' enviornment at 300 baud but NOT 'on-line' with a database or BBS. You may try it to see for your self but you will have a screen full of half-written words, lost characters and who knows what. BASIC is TOO SLOW even when the program is in it's simplest of forms and you can't get much simpler than the EXAMPLE routine on page 6. There is NO cursor, there is only 'checking' done for characters LESS-THAN an ASCII 32 (space). CARRAGE-RETURN (ascii-13) is not even tested. All of which are essential items even for a 'dumb' terminal routine. BASIC is not the best choice for communications on the Timex machines. Ah but there IS a way to make BASIC faster COMPILE IT!

On side 'B' there are two versions of the EXAMPLE program on page 6 in the Z-SI/O manual that have been compiled with the TIMACHINE compiler by NovelSoft. I did not add any 'checking' other than for carriage return (ascii-13) and made the program fully menu driven. To load the machine code version type LOAD"DUMBTERM" The two versions (HALF DUPLEX & FULL DUPLEX) are the same program compiled twice and placed 'back to back'. I kept it SIMPLE and as close to the LLISTing in the manual as possible. There are 2 commands available when in TERMINAL MODE. A 'CAP SHIFT - 1' (EDIT) will return you to the 'LOCAL CONTROL MENU'. A 'SYMBOL SHIFT - A' (STOP) will return you to the BASIC loader (HALF/FULL DUPLEX) select menu. To load the 'source' file that was used by the TIMACHINE compiler type LOAD "ZDTSOURCE". If you have TIMACHINE you could add your own routines like a true DELETE and a 'PRINT AT x,y' for a flashing cursor and more. The TIMACHINE by NovelSoft is my favorite because of it's speed, features and ease of use. But there are others that could do as good a job. A compiler is ok for simple routines but they have their limits. The ideal route is machine code and a good assembler is a must. There are several and are really up to the user which one he or she likes best. Some of the best known are ZUES, HOT-Z and one from the U.K. by PICTURESQUE (my favorite).

WHY RE-INVENT THE WHEEL? There is ALLREADY an excellent telecommunication software package for the Z-SI/O card. It's features include 64 col display, XMODEM protocol for file transfers and 300/1200 baud support for use with ANY RS-232 'stand-alone' modem. For more information contact:

GREY & CLIFFORD COMPUTER PRODUCTS
P.O.Box #2186
Inglewood, CA 90303
(213) 759-7406